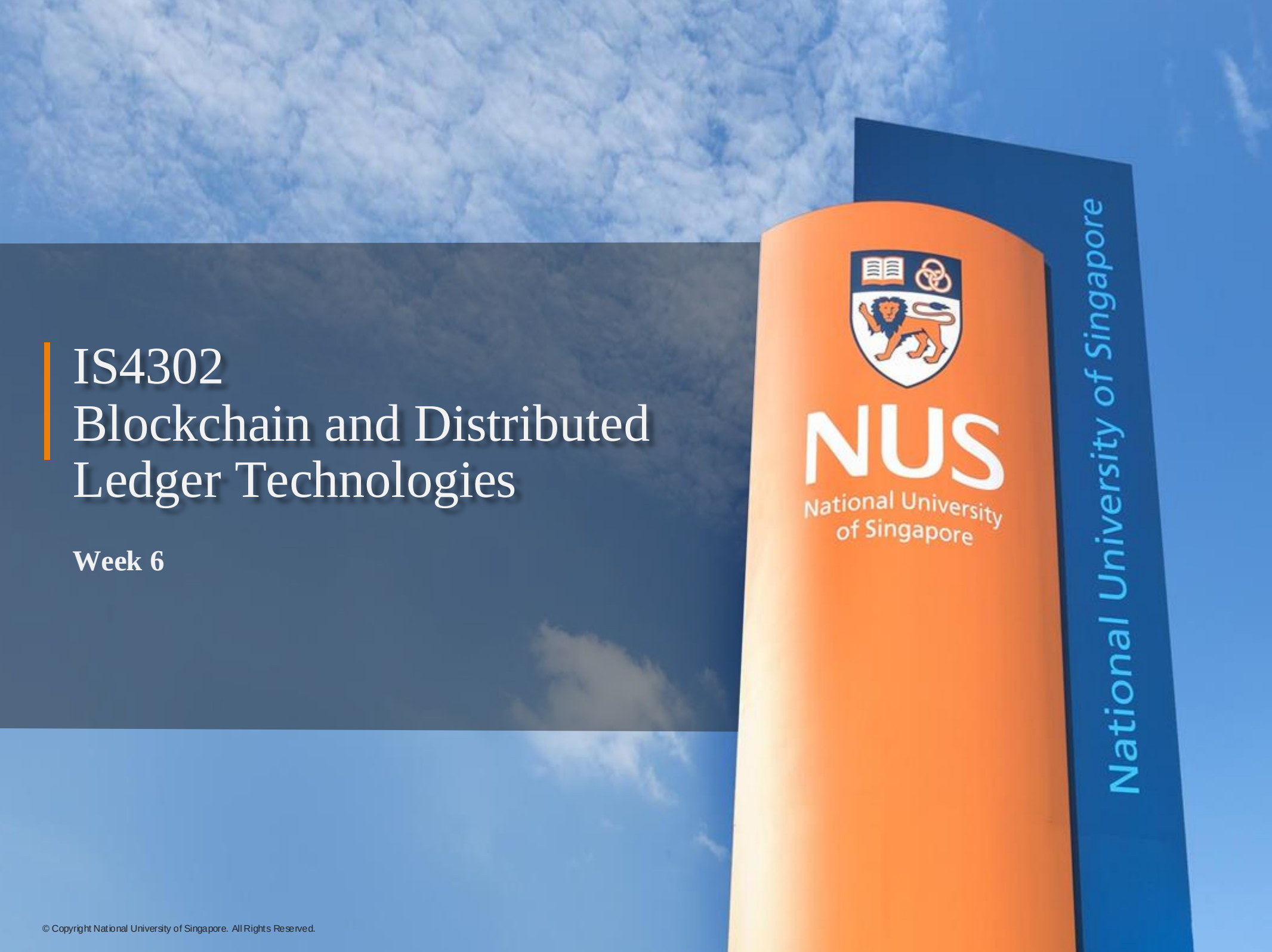


IS4302 Blockchain and Distributed Ledger Technologies

Week 6

Agenda

- **Project Overview**
  - Group project deliverables
  - Assumptions and design goals
- **Use Cases and Solutions**
  - Tokenization of IP
  - Track & Trace
- **Oracle Problem and Blockchain Oracles**
- **Example dApp Deep-Dive**
  - Public Procurement
- **Reflections and Key Takeaways**

Intended Learning Outcomes

1. **Understand the differences between traditional web applications and decentralized apps (dApps).**
2. **Gain insights into tokenization applications, especially for intellectual property (IP).**
3. **Learn the role and challenges of oracles in blockchain ecosystems.**
4. **Explore how smart contracts can streamline processes like public procurement and delivery verification.**
5. **Reflect on the limitations and innovative potential of blockchain solutions in real-world scenarios.**

Agenda

- **Project Overview**
  - Group project deliverables
  - Assumptions and design goals
- **Use Cases and Solutions**
  - Tokenization of IP
  - Track & Trace
- **Oracle Problem and Blockchain Oracles**
- **Example dApp Deep-Dive**
  - Public Procurement
- **Reflections and Key Takeaways**

Group project

- **Deliverables**
  - Demo presentation / walkthrough
    - **(submit a screen-grab recorded video)**
    - **No frontend required**
  - Architecture and design document
    - **Business logic and technical implementation**
  - Code
    - **As a GitHub link in the document**
  - Survey on peer evaluation

Group project criteria

- **Don’t need to be extremely innovative in terms of industry.**

- **Be practical, be specific, be comprehensive.**
  - Think about each stakeholder’s incentive, information structure,…
  - Try to design a system such that it makes sense.
  - Don’t be afraid of limitations, but try to make them fall in the least critical aspects
  - Be innovative at overcoming limitations
  - Discuss those limitations

Assumptions

- **Reasonable assumptions beyond current practice can be made.**
  - E.g., Cost for pure computation can be low (introduction of cloud service and layer-2s)
  - The existence of reasonable/incentive compatible oracle systems can be assumed.

What to think about

- **To B? To C?**
- **Why does each stakeholder want to use your system?**
- **Will they behave as intended?**
- **What does each stakeholder know/could know? Is this desirable?**

What to think about

- **Is a token needed?**
- **If so, how to design the tokenomics?**
  - Exclusivity?
  - Limited supply or constant supply or something else?
  - Speculation needed or not?
- **Is data segregation needed? If so, how?**
- **……**

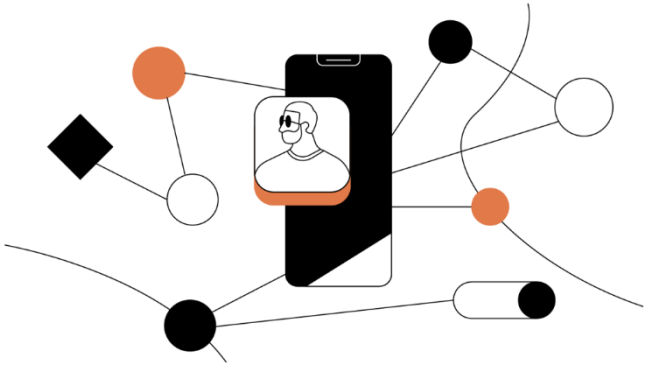
Agenda

- **Project Overview**
  - Group project deliverables
  - Assumptions and design goals
- **Use Cases and Solutions**
  - Tokenization of IP
  - Track & Trace
- **Oracle Problem and Blockchain Oracles**
- **Example dApp Deep-Dive**
  - Public Procurement
- **Reflections and Key Takeaways**

What Are dApps?

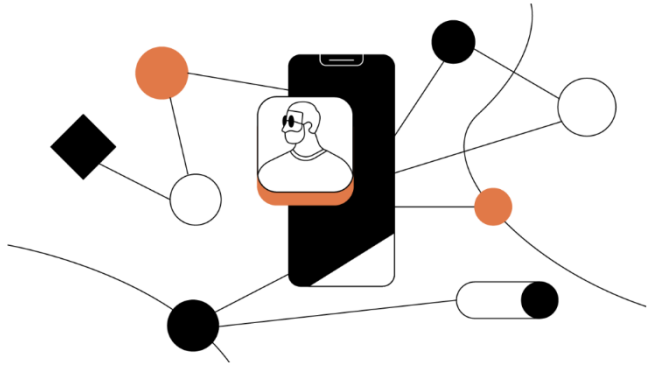
Decentralized applications (dApps):

- Run on top of blockchain networks.
- Similar to web applications in terms of user experience (UX)
- Their back-end processes differ:
  - dApps avoid centralized servers
  - They transact in a distributed and peer-to-peer (P2P) fashion
  - They do not use the central HTTP protocol to communicate.



What Are dApps?

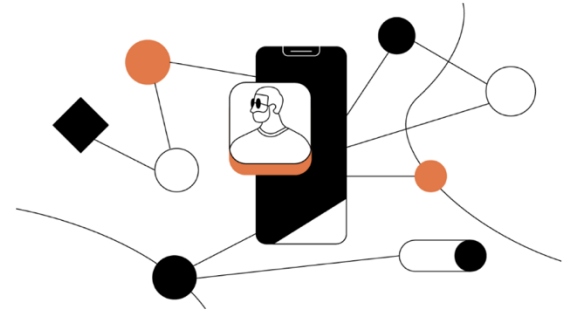
- **Most dApps are built with Ethereum**
- **All dApps are built using smart contracts.**
  - Smart contracts:
    - **Automated, self-executing programs**
    - **Hosted on a decentralized "virtual machine", such as Ethereum's EVM**
    - **They make transactions between two parties seamless, quick, and automatic.**



Traditional Web Apps vs. Decentralized Apps

Traditional Web Applications:

- Software or programs that run on a web server rather than a device's operating system
  - Dropbox, Slack, and Twitter are examples of companies that offer web applications
- Devices and servers communicate via coding messages through the Hypertext Transfer Protocol (HTTP).
- The feed on display (the front end) is drawn from data held on the company's web server (the back end).



Traditional Web Apps vs. Decentralized Apps

Decentralized Web Applications:

- A dApp has a front-end user interface (UI) that communicates with smart contracts that transact on the blockchain
- On the back end, dApps communicate with their respective blockchain networks through a wallet, which serves as a bridge to the blockchain ecosystem.

Traditional Web Apps vs. Decentralized Apps

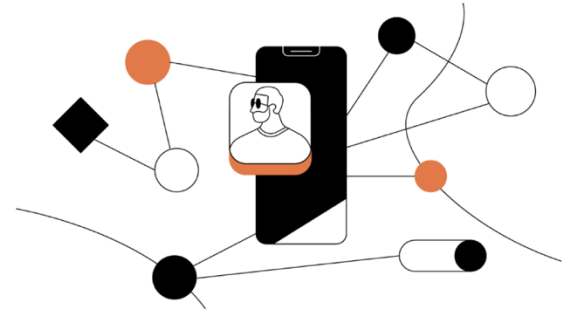
Some Examples:

Lending and borrowing

Aave
Lend your tokens to earn interest and withdraw any time.

Browsers

Brave
Earn tokens for browsing and support your favorite creators with them.



Utilities

IPFS
A peer-to-peer hypermedia protocol designed to preserve and grow humanity's knowledge by making the web upgradeable, resilient, and more open.

Social media

GM
All-in-one platform for chat, forums, and voice that actually shares revenue with its creators

Explore more here:
https://ethereum.org/dapps

Digital Assets aka Tokens

- **Definition:**
  - representation of information on a blockchain that confers ownership, access rights, representation, voting/utility rights
- **Types:**
  - Payment tokens (digital currencies)
  - Utility tokens (governance and access)
  - Security tokens (equity, debt, financial assets)

Digital Assets

- **Non-financial use cases:**
  - supply chain, identity management, data storage

- **Financial use cases:**
  - tokenised securities, fundraising, payment infrastructure

Agenda

- **Project Overview**
  - Group project deliverables
  - Assumptions and design goals
- **Use Cases and Solutions**
  - Tokenization of IP
  - Track & Trace
- **Oracle Problem and Blockchain Oracles**
- **Example dApp Deep-Dive**
  - Public Procurement
- **Reflections and Key Takeaways**

Tokenisation of IP

- **Intellectual Property (IP)**
  - Rights over creations of the human mind
  - Sanctioned by state authority
  - Facilitate exploitation by creator
- **Purpose of IP:**
  - Prevent free-riders from profiting without consent
  - Encourage innovation and product differentiation

Tokenisation of IP

- **Types of IP:**
  - Patents (protection of inventions)
  - Copyright (protection of works of authorship)
  - Trademarks (prohibition of unauthorised usage of unique signs)
- **Role of Rights Holder:**
  - Enables profit for rightsholder
  - Rights may be transferred to investors for mutually beneficial transactions

Tokenisation of IP

- **Challenges in IP Management:**
  - Manual management across different systems (e.g., paper, ERP systems, digital documents)
  - Friction and efficiency loss due to media discontinuity
  - Difficulties in tracking updates and sharing across parties

Tokenisation of IP

- **Tokens as Digital Representations of IP Rights:**
  - Content creators can sell IP tokens for upfront funding
  - Tokens can be transferred and new earnings distributed continually through smart contracts

- **Benefits of Tokenisation of IP:**
  - Streamlines management and reduces friction
  - Enables faster proof of origin
  - Facilitates funding and investment opportunities

Tokenisation of IP

- **Example: *Bernstein.io***
  - Allows content creators to document IP-based digital assets
  - Register on the Bitcoin blockchain for unique and verifiable provenance
  - Timestamping by EU and China central authorities for official recognition

Tokenisation of IP

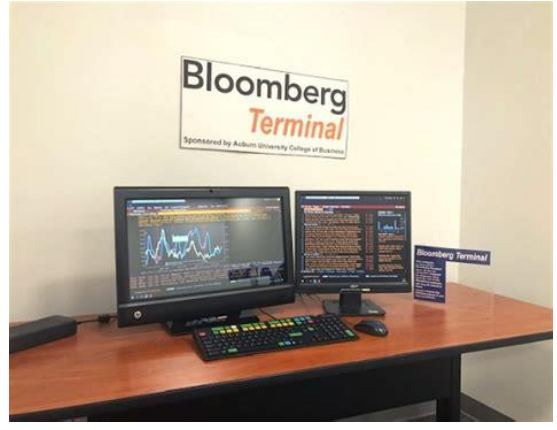
- **Representation as NFTs**
  - Potential for digitised IP assets to become Non-Fungible Tokens (NFTs)
  - Increased liquidity and potential for trading or renting on different blockchain systems
  - NFTs can provide beneficiaries with ways to leverage their image for a larger revenue stream
- **Examples:**
  - IPCG for tokenizing patents
  - Lexit for tokenizing “metaverse” IP and assets

Licensing

- **Issues with Software Licensing**
  - Challenges in predicting usage and determining proper number of licenses to purchase
  - Operational problems for software companies tracking license validity and user correspondence
  - Limited consumer base due to perceived high cost and illiquidity of licenses

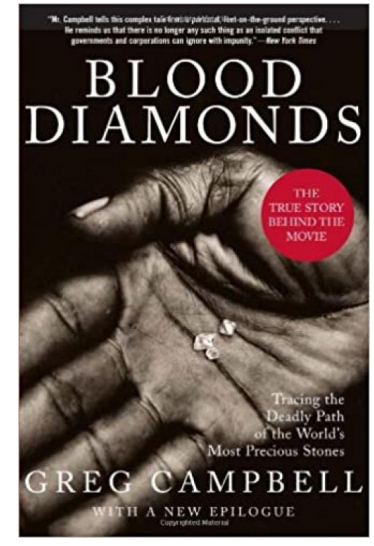
Token-As-A-License (TaaL) Model

- **Identifying each software license with a token**
- **Validity of used licenses can be easily verified against the blockchain**
- **Solves double-spending problem**
- **Enables secondary market for used licenses**



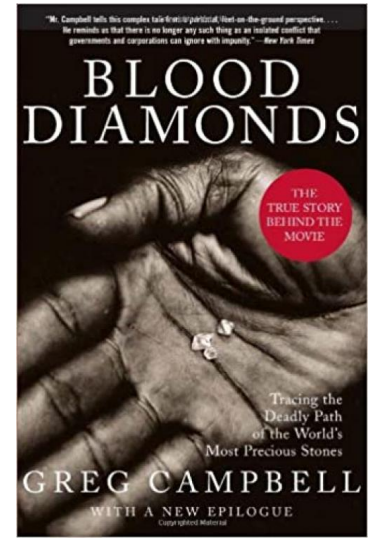
Track & trace

- **Quality, ecological and ethical standards is important for consumers and public authorities**
- **Proving standards are met is difficult due to the complexity of supply chains**
- **Product traceability is crucial for accountability and assurance**



Track & trace

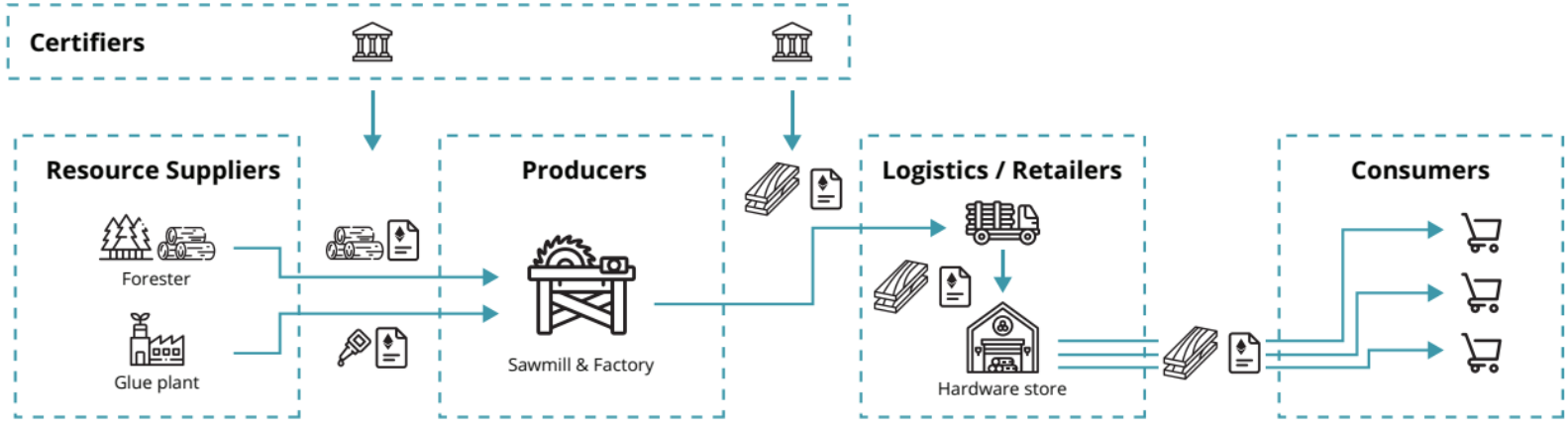
- **Challenges of Product Traceability**
  - Information stored in isolated systems
  - Lack of communication and trust between systems
  - Difficulty in identifying and tracking products/components in real-time
  - Limited transparency and traceability throughout the supply chain



Track & trace

Tokenisation Solution

- Non-fungible tokens (NFTs) can be created for every item or batch of items
- Mirrors each step of the production chain on the blockchain
- Enables granular traceability and real-time tracking of products/components



Track & trace

Benefits of Tokenisation for Supply Chain Management

- Identifies exact path of each item
- Provides information on geographical origin, issuance date and transaction data
- Enables organisations to understand and control product life-cycle
- Third-party certifiers can be integrated into the supply chain underpinned by tokenisation
- Integration with IoT devices provides real-time data on products/components

Track & trace

Benefits of Tokenisation for Supply Chain Management

- Identifies exact path of each item
- Provides information on geographical origin, issuance date and transaction data
- Enables organisations to understand and control product life-cycle
- Third-party certifiers can be integrated into the supply chain underpinned by tokenisation
- Integration with IoT devices provides real-time data on products/components

Agenda

- **Project Overview**
  - Group project deliverables
  - Assumptions and design goals
- **Use Cases and Solutions**
  - Tokenization of IP
  - Track & Trace
- **Oracle Problem and Blockchain Oracles**
- **Example dApp Deep-Dive**
  - Public Procurement
- **Reflections and Key Takeaways**

Oracle Problem -- Overview

- **The oracle problem**
  - Reference: Caldarelli, Giulio. "Understanding the blockchain oracle problem: A call for action." Information 11.11 (2020): 509.

- **Oracles in the current ecosystem**
  - Reference: https://chain.link/

- **Example of a decentralized oracle**
  - Reference: Adler, John, et al. "Astraea: A decentralized blockchain oracle." 2018 IEEE international conference on internet of things (IThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData). IEEE, 2018.

What Are Oracles?

- **The term “oracle” comes from Greek mythology**
  - refers to someone able to communicate directly with god and see the future
- **In ancient stories, people did not have enough information to make decisions, and turned to oracles for knowledge beyond their understanding.**
- **In the blockchain environment, oracles are systems that provide blockchain with information coming from the real world.**

What Are Oracles?

- **In a blockchain, extrinsic information cannot be provided along with transaction data, since other nodes would detect information coming from an “untrusted” source.**
- **Information coming from the real world should come from a third-party univocal source**
- **Reliability is undisputed for all nodes**

- **Oracles (on the blockchain) do not predict the future but retrieve information from the past**

What Are Oracles

- **Oracles are not specific programs or devices, but "concepts".**
  - Anything providing external data to the blockchain can be classified as an oracle.
- **Oracles, in general, do not insert information into the blockchain directly; conversely, they gather and store data from the real world.**
- **When a smart contract concerning extrinsic data is executed, the code then calls for the right information from a trusted oracle.**

Examples of oracles

- **IoT systems such as probes and sensors**
- **Platforms such as ERP**
- **In the case of private data, the very human that operates directly on the blockchain**

Examples of oracles

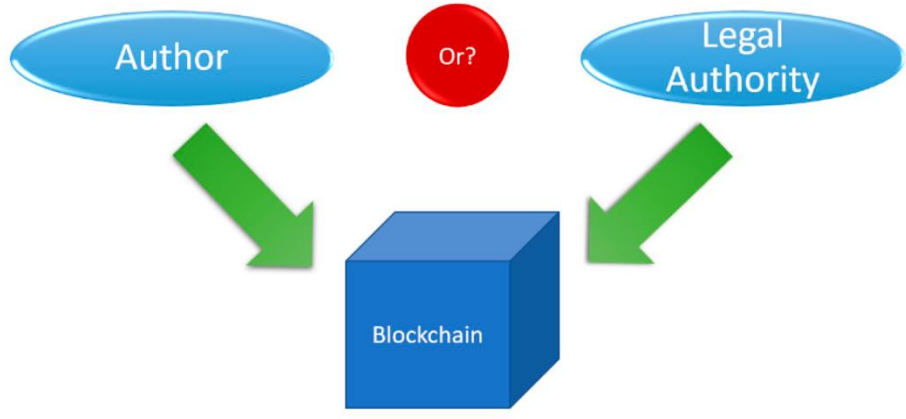
- **Examples of data gathered by oracles**
  - Events/data in other blockchains
  - Randomness
  - Lottery winners
  - Natural disasters along with risk measurements
  - Price and exchange rate of real/crypto assets
  - Static data (e.g., country codes)
  - Dynamic data (e.g., time measurements)
  - Weather conditions
  - Political events
  - Sporting events
  - …

The oracle problem

- **The security, authenticity, and trust conflict between third-party oracles and the trustless execution of smart contracts**
  - Reintroduce the single-point-of-failure
  - Needs to be trusted, removing trustless peer-to-peer interaction
  - “two step-back from decentralization”

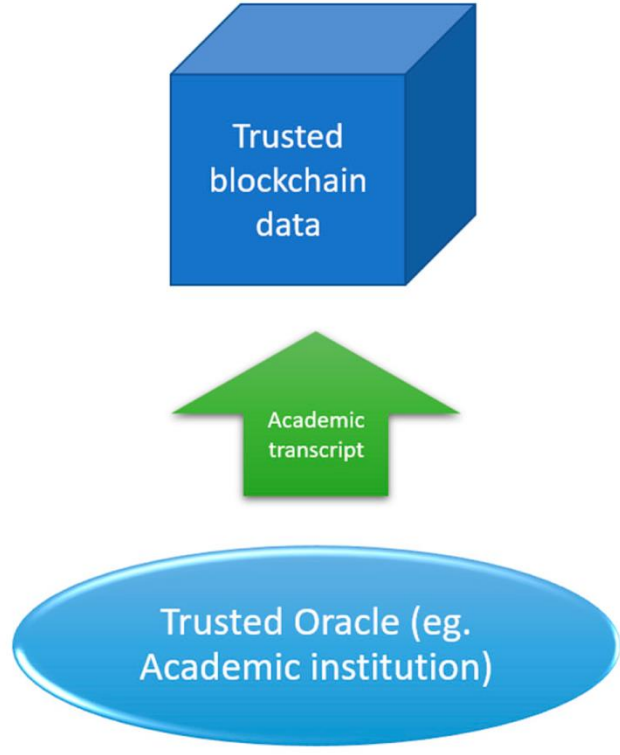
Example – Intellectual Property Rights (IPRs) protection

- **The original purpose of the blockchain, before the crypto 'era', was to "register" intellectual property rights**
  - Protect against tampering
  - Efficiently share revenues
  - Eliminate intermediaries?
- **Who decides ownership? Author or legal authorization?**
  - What if someone recognizing his or her work as registered by someone else?



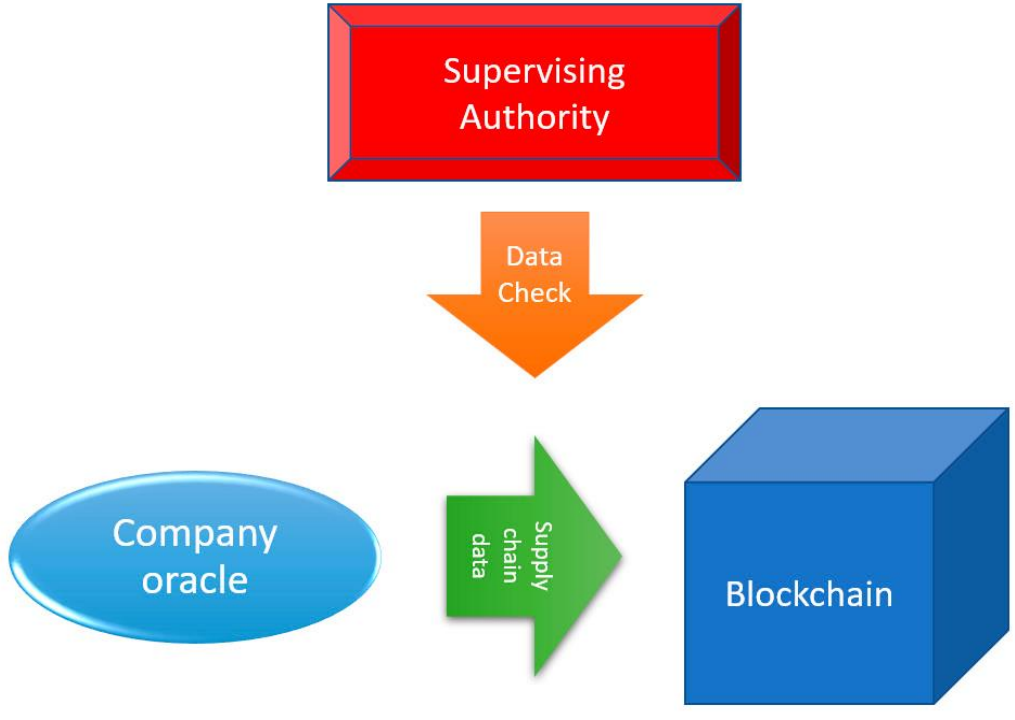
Example – Academic Transcript

- **The universities are themselves Oracles**
  - Universities, especially reputable ones, are generally trustworthy



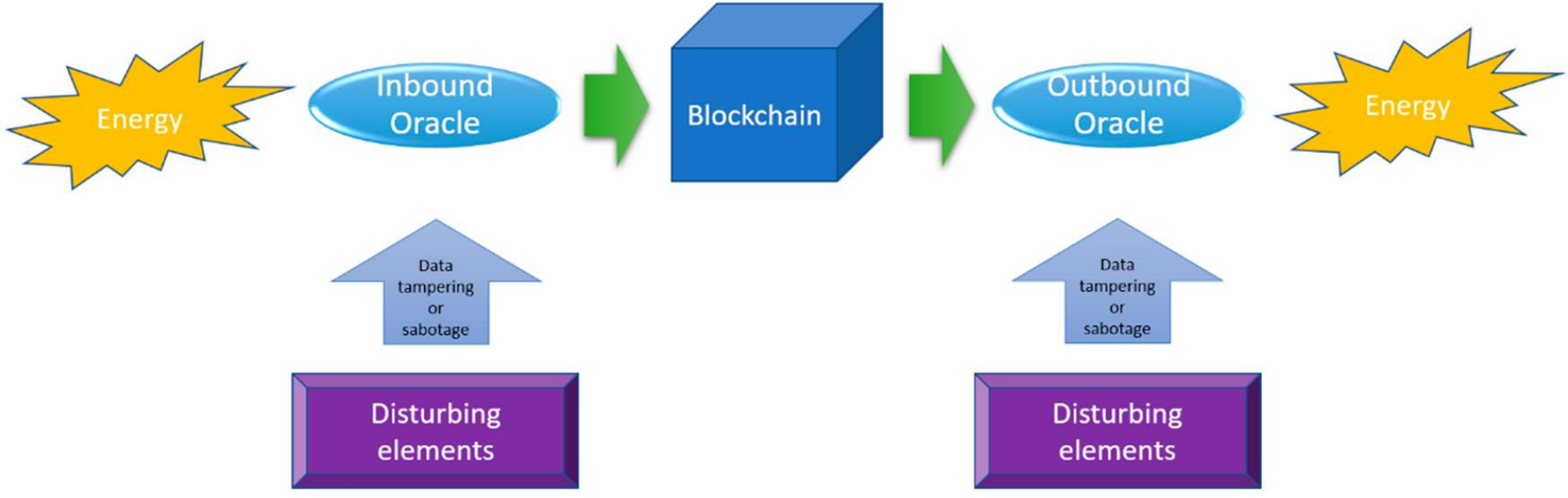
Example – Supply chain management

- **Oracles belong to the company producing goods that are being tracked**
  - Companies decide what information to upload on the blockchain
- **An auditor?**
  - Who audits the auditor?
  - Does benefits outweighs costs?



Example - Energy

- **Introducing blockchain within the energy sector**
  - reduction in costs for the marketplace
  - increased transparency and decentralization



Example - Energy

- **The platform could be decentralized and independent from a central authority**
- **Oracles are unlikely to also be decentralized and autonomous**
  - Local infrastructure
  - Monitoring and maintenance
  - …

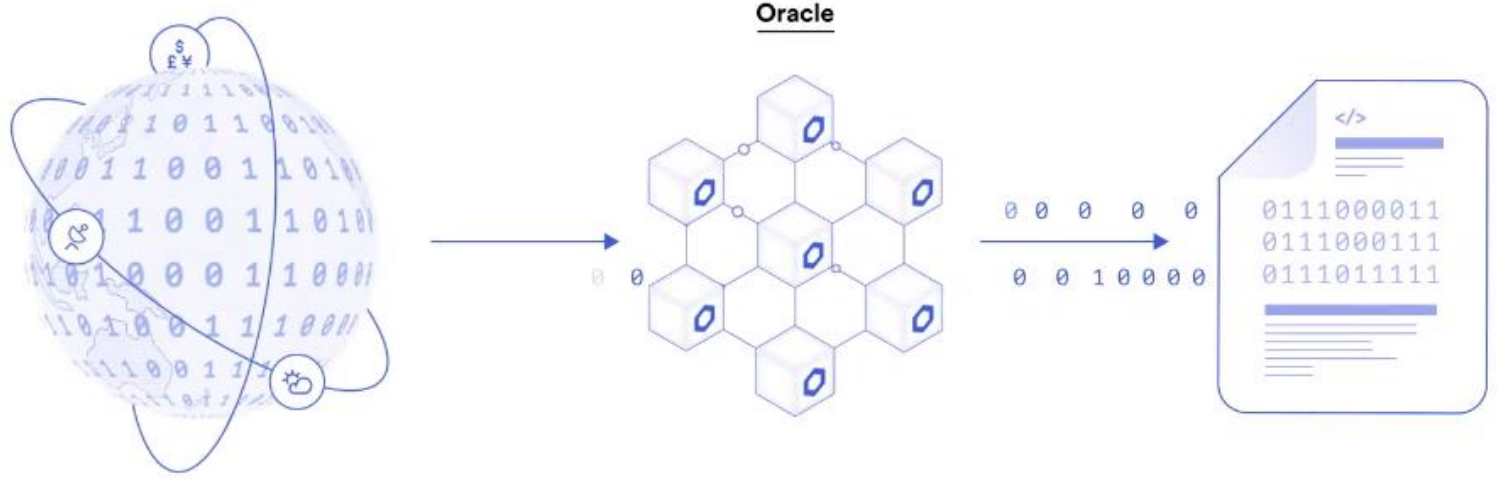
Overview

- **The oracle problem**
  - Reference: Caldarelli, Giulio. "Understanding the blockchain oracle problem: A call for action." Information 11.11 (2020): 509.

- **Oracles in the current ecosystem**
  - Reference: https://chain.link/

- **Example of a decentralized oracle**
  - Reference: Adler, John, et al. "Astraea: A decentralized blockchain oracle." 2018 IEEE international conference on internet of things (IThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData). IEEE, 2018.

Chainlink Overview



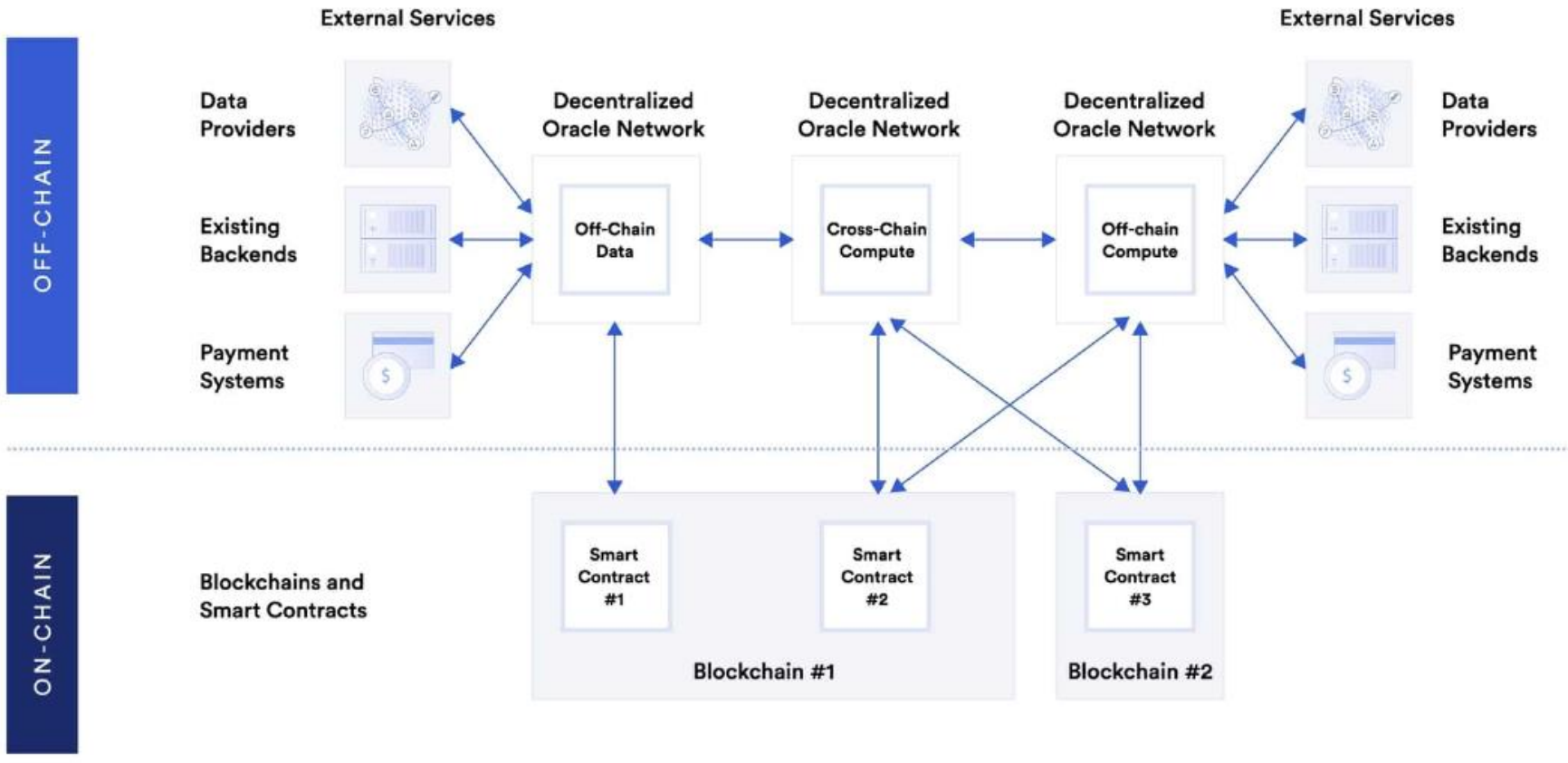
https://youtu.be/GnXsJe2wZ_w?si=magQAbKoFUYeio7w&t=357

Types of Blockchain Oracles

- **Input Oracles**
  - most widely recognized type of oracle
  - fetches data from the real-world (off-chain) and delivers it onto a blockchain network for smart contract consumption
- **Output Oracles**
  - allow smart contracts to send commands to off-chain systems that trigger them to execute certain actions
  - E.g., informing a banking network to make a payment
  - telling a storage provider to store the supplied data
  - pinging an IoT system to unlock a car door once the on-chain rental payment is made

Types of Blockchain Oracles

- **Cross-Chain Oracles**
  - read and write information between different blockchains
  - enable interoperability for moving both data and assets between blockchains
  - E.g., using data on one blockchain to trigger an action on another or bridging assets cross-chain
- **Compute-Enabled Oracles**
  - secure off-chain computation to provide services that are impractical to do on-chain due to technical, legal, or financial constraints
  - E.g., computing zero-knowledge proofs to generate data privacy, running a verifiable randomness function



Different types of oracles enable the creation of hybrid smart contracts

Blockchain Oracle Use Cases

- **DeFi**
  - decentralized money markets use price oracles to determine users' borrowing capacity and check if users' positions are undercollateralized and subject to liquidation
  - automated market makers (AMMs) use price oracles to help concentrate liquidity at the current market price to improve capital efficiency
  - …

Blockchain Oracle Use Cases

- **Dynamic NFTs and Gaming**
  - Dynamic NFTs—Non-Fungible Tokens that can change in appearance, value, or distribution based on external events like the time of day or the weather
  - Compute oracles are used to generate verifiable randomness that projects then use to assign randomized traits to NFTs or to select random lucky winners in high-demand NFT drops.
  - On-chain gaming applications also use verifiable randomness

Blockchain Oracle Use Cases

- **Insurance**
  - Insurance smart contracts use input oracles to verify the occurrence of insurable events during claims processing, opening up access to physical sensors, web APIs, satellite imagery, and legal data.
- …

Overview

- **The oracle problem**
  - Reference: Caldarelli, Giulio. "Understanding the blockchain oracle problem: A call for action." Information 11.11 (2020): 509.

- **Oracles in the current ecosystem**
  - Reference: https://chain.link/

Potential applications

- **Data Availability**
- **Machine Learning and Data Annotation**
- **Adjudication Mechanisms**

Agenda

- **Project Overview**
  - Group project deliverables
  - Assumptions and design goals
- **Use Cases and Solutions**
  - Tokenization of IP
  - Track & Trace
- **Oracle Problem and Blockchain Oracles**
- **Example dApp Deep-Dive**
  - Public Procurement
- **Reflections and Key Takeaways**

Overview

Application Areas in Non-Financial Service Sectors

Public Procurement and Corruption

- **World Economic Forum:**
  - *"**Public sector corruption** is the single largest challenge, stifling social, economic, and environmental development. Often, corruption centers around a lack of transparency, inaccurate record keeping, and low public accountability."*

Public Procurement and Corruption

- **World Economic Forum:**
  - *"**Public sector corruption** is the single largest challenge, stifling social, economic, and environmental development. Often, corruption centers around a lack of transparency, inaccurate record keeping, and low public accountability."*
  - *"Corruption severely and disproportionally affect the poorest and most vulnerable in any society,…, deters investment, weakens economic growth and undermines the basis for law and order."*

Public Procurement and Corruption

- **World Economic Forum:**
  - *"**Public sector corruption** is the single largest challenge, stifling social, economic, and environmental development. Often, corruption centers around a **lack of transparency**, **inaccurate record keeping**, and **low public accountability**."*
  - *"Corruption severely and disproportionally affect the poorest and most vulnerable in any society,…, deters investment, weakens economic growth and undermines the basis for law and order."*

Public Procurement Process

- **Defined as:** “the process of identifying **what** is needed; determining **who** the best person or organization is to supply this need; and **ensuring** what is needed is delivered to the **right place**, at the **right time**, for the **best price** and that all this is done in a **fair and open** manner”

Public Procurement Process

- **Defined as:** “the process of identifying **what** is needed; determining **who** the best person or organization is to supply this need; and **ensuring** what is needed is delivered to the **right place**, at the **right time**, for the **best price** and that all this is done in a **fair and open** manner”

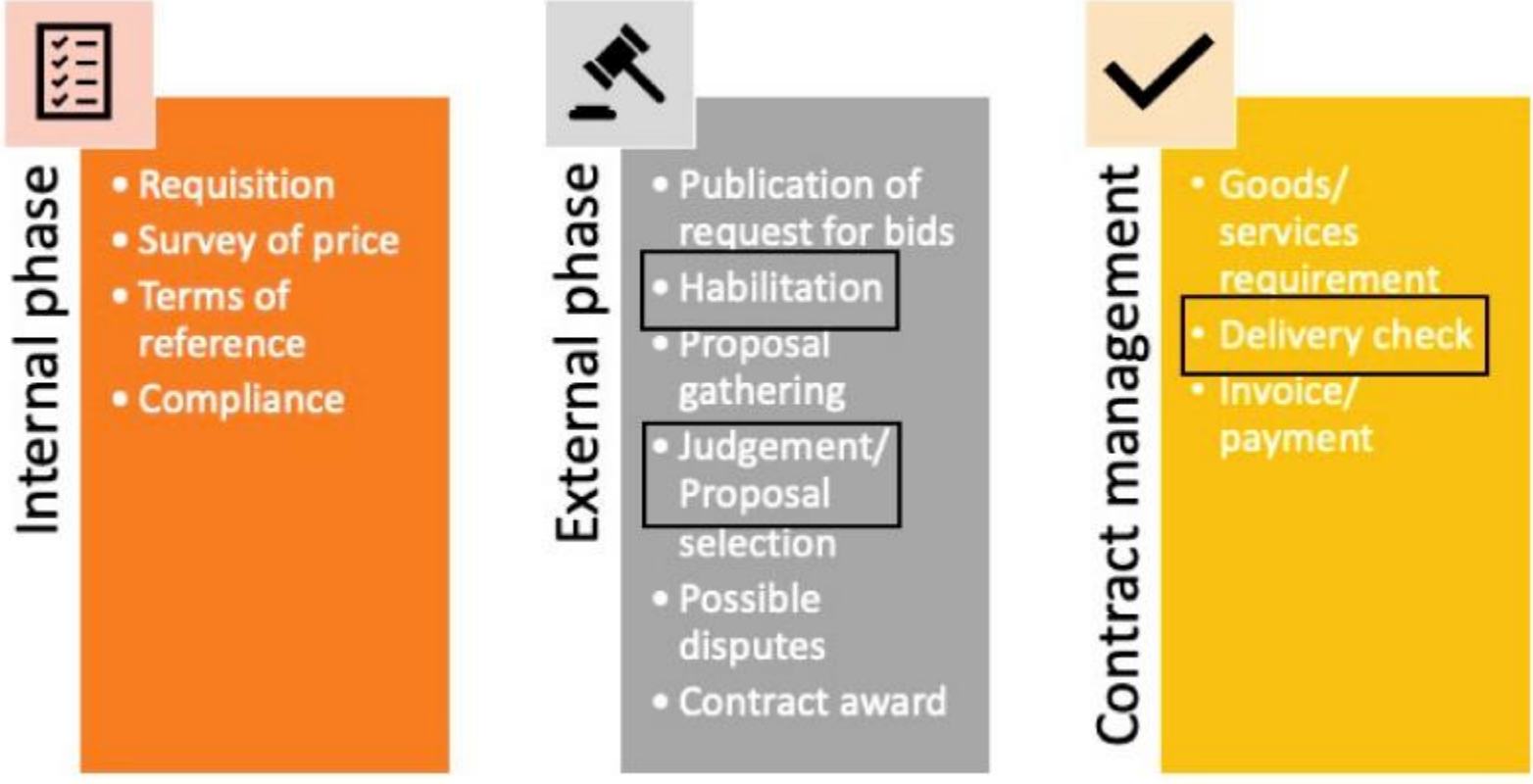
- **Ideal features:** transparency, integrity, access, balance, participation, efficiency, e-procurement, capacity, evaluation, risk management, accountability and integration.

Public Procurement Process

Three Basic Stages:

1. **Internal Phase:** Sets the terms of the contract to be bid on, the rules of the bidding process, and which public agents will be involved
2. **External Phase:** Publication of contracts to promote fair competition. Judging process of the bids.
3. **Integration:** The winning contractor is then supervised until completion of the work.

Public Procurement Process



Public Procurement Process

Three Basic Stages:

1. **Internal Phase:** Sets the terms of the contract to be bid on, the rules of the bidding process, and which public agents will be involved
   - This phase is seen as the *most* prone to corruption
     - **Requirements and needs assessment are crafted to benefit specific contractors.**
     - **Informal, illegal agreements between government workers and contractors**
     - **Undue disclosure of information (e.g. competitor's prices)**

Public Procurement Process

- **External Phase:** Publication of contracts to promote fair competition. Judging process of the bids.
  - Sources of Corruption:
    - **Not enough time/visibility for all contractors to see a RFP**
    - **Choosing a worse bid**
- **Integration:** The winning contractor is then supervised until completion of the work.
  - Sources of Corruption:
    - **Acceptance of damaged or forged goods**
    - **Approval of products different from those specified in contract**

Smart Contract Solution

- **Being pursued in countries such as: Mexico, Seoul South Korea, Colombia, South Africa, Denmark, and Brazil**

- **Our Particular example:**
  - Create Smart Contracts that Focus on:
    - **Bidding**
    - **Supplier habilitation**
    - **Delivery Verification**

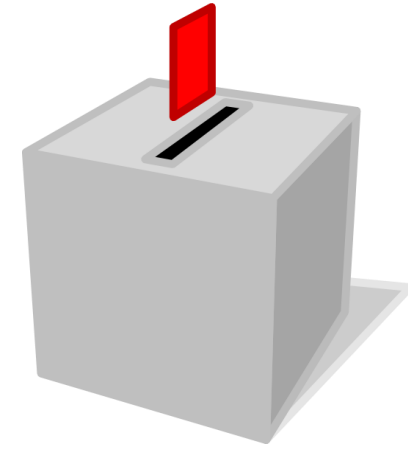
Smart Contract Solution

- **Stages of the Tendering Process**
  - Tender: process whereby governments and financial institutions invite bids for large projects that must be submitted within a finite deadline.

Smart Contract Solution

- **Bidding:**
  - Assume the cheapest offer wins
  - Prevent disclosure of competing offers
    - **Do not transmit or publicly store price info in the contract**
  - Do not allow companies to change bids once submitted
  - Prevent Spam offers

Smart Contract Solution



- **Bidding Pseudo Code:**
  - Bidding hash: hash of the price value of the bid (uses an nonce/salt as well for privacy)
  - Offer hash: hash of the contract (offer document)
  - Tender: The Government Entity's Contract's Public Key
  - Sender: Contractor's Public Key

Algorithm 1: *Smart Contract Place Bid*—places bid and stores the hashes of the bid and offer document in the smart contract

```
1 input: bidding hash, offer hash, tender, sender
2 if status of tender is propose (tender) AND not duplicate bid (sender)
3     store (bidding hash)
4     store (offer hash)
5     lock deposit (sender)
6     register bid (sender)
7 end if
```

Smart Contract Solution

- **Reveal Pseudo Code:**
  - Vendors message the smart contract with their price and the nonce to verify their original bids

Algorithm 2: *Smart Contract Reveal Bid*—reveals the bid after the offering period

```
1  input: price, nonce, tender, sender
2  if status of tender is reveal (tender)
3     if bidding hash = hash (price, nonce)
4        add price (price, sender)
5        if all vendors revealed OR revealing period exceeded
6              verify and mark perliminary winner
7        end if
8     else error: price or/and nonce are incorrect
9     end if
10 end if
```

Smart Contract Solution

- **Supplier Habilitation**
  - Contractors must be checked for compliance:
    - **Taxes, working conditions, quality seals, permission to manufacture medical devices, etc.**
  - Sovereign Identity for contractors
    - **Contractors control their own unique ID**
    - **Certifications from an authorized body can be stored on a public blockchain for each vendor.**
  - A smart contract can automatically check that all certificates are in place.
    - **Prevents manual manipulation**

Smart Contract Solution

- **Supplier Habilitation**
  - Pseudo Code

Algorithm 3: *Smart Contract Supplier Habilitation of the Winner*—verifies that the winner of the bidding meets all requirements and places the order

```
1  input: winner, tender
2  if status of tender is reveal (tender)
3     if all requirements met (winner)
4        winner approved (winner)
5        place order (winner)
6        lock payment (getPrice (winner))
7     else
8        mark next winner (tender)
9        call supplier habilitation of the winner
10    end if
11 end if
```

Smart Contract Solution

- **Delivery Verification**
  - Human oracle is needed.
    - **Why?**

Smart Contract Solution

- **Delivery Verification**
  - Human oracle is needed.
    - **To reduce “single point of failure” the smart contract chooses randomly 2 (or more) auditors from a pre-defined set.**
    - **When all auditors approve the delivery, payment is triggered**
    - **If there is a dispute, external arbitration is triggered**

Smart Contract Solution

- **Delivery Verification**
  - Pseudocode

Algorithm 4: *Smart Contract Order Verification*—verifies the order and releases payment

```
1  input: tender
2  prerequisite: only buying party can execute
3  if status of tender is evaluate (tender)
4     repeat
5        select auditor randomly
6        wait for validation of auditor
7     until number of validations reached
8     if all validations correct
9        release payment
10    else
11       start arbitration process
12    end if
13 end if
```

Reflect

Q: What might be some cons of using the above system?

Reflect

Q2: What are the potential benefits?

Conclusion

Blockchain Benefits

- **Decentralization and transparency improve efficiency and reduce fraud.**
- **dApps and smart contracts automate processes, removing intermediaries.**

Real-World Applications

- **Blockchain use cases like IP tokenization and public procurement streamline operations.**
- **Smart contracts ensure compliance, accuracy, and transparency.**

Key Takeaways

Oracle Challenges

- **Oracles connect blockchain with real-world data but create trust and security issues.**
- **Decentralized oracles and random auditor systems offer solutions to mitigate these risks.**

Thank you!