Ref. Ares(2021)7550445 - 07/12/2021



ICT-56-2020 "Next Generation Internet of Things"
Grant Agreement number: 957218

IntellioT

# Deliverable D3.4
# Decentralized trust via secure interaction and contracts (first version)

| | |
|---|---|
| Deliverable release date | 30/11/2021 |
| Authors | 1. AAU: Lam Nguyen<br>2. AAU: Beatriz Soret<br>3. SANL: Konstantinos Fysarakis, Ioannis Vezakis<br>4. TSI: Charalampos Savvakos |
| Editor | Lam Duc Nguyen (AAU), Beatriz Soret (AAU) |
| Reviewer | Anis El Gabli, Sumudu Samarakoon (UOULU), Konstantinos Fysarakis (SANL) |
| Approved by | PTC Members: (Vivek Kulkarni, Konstantinos Fysarakis, Sumudu Samarakoon, Beatriz Soret, Arne Bröring, Maren Lesche)<br>PCC Members: (Vivek Kulkarni, Jérôme Härri, Beatriz Soret, Mehdi Bennis, Martijn Rooker, Sotiris Ioannidis, Anca Bucur, Georgios Spanoudakis, Simon Mayer, Filippo Leddi, Harshitha Chandregowda, Maren Lesche, Georgios Kochiadakis) |
| Status of the Document | Final |
| Version | 1.0 |
| Dissemination level | Public |

# Table of Contents

1 Introduction ..... 3
1.1 Overview of IntellIoT framework ..... 3
1.2 Deliverable Outline ..... 4
2 Background ..... 5
2.1 Distributed ledger Technologies ..... 5
2.2 Transaction Validation ..... 6
2.3 DLTs Comparison ..... 6
3 DLT-Based Communication protocols and storage ..... 9
3.1 DLT-based communication protocols in resource-constrained IoT ..... 9
3.2 Distributed Data Storage ..... 11
3.3 IntellIoT's different DLT Implementation Perspectives ..... 12
3.3.1 DLT for Agriculture ..... 12
3.3.2 DLT for Healthcare ..... 14
3.3.3 DLT for Manufacturing ..... 16
3.3.4 DLT in IntellIoT's Trustworthiness pillar ..... 18
4 DLT assisted distributed learning ..... 20
4.1 Overview ..... 20
4.2 DLT-based infrastructure for Federated Learning ..... 21
4.3 Freshness and valuation of data ..... 22
5 Conclusion & Future Direction ..... 25
References ..... 26

# 1 INTRODUCTION

## 1.1 Overview of IntellIoT framework

One of the pillars of the IntellIoT framework is trustworthiness (see the project's architecture overview in Figure 1, as defined in D2.3 ("High level architecture"). The security and privacy enablers are addressed in WP4 ("Efficient, reliable and trustworthy computation & communication infrastructure"), specifically in Task 4.4 ("Trustworthy infrastructure by design"). The third component of the trustworthiness, **Distributed Ledger Technologies (DLT) and smart contracts**, is specific of distributed systems and collaborative IoT, and therefore addressed in this Task 3.4 ("Decentralized trust via secure interaction & contracts"") belonging to WP3 ("Distributed, self-aware IoT applications & human-defined autonomy"). This deliverable D3.4 ("Decentralized trust via secure interaction and contracts") aims to describe the first version of the research and implementation tasks related to DLT and smart contracts.

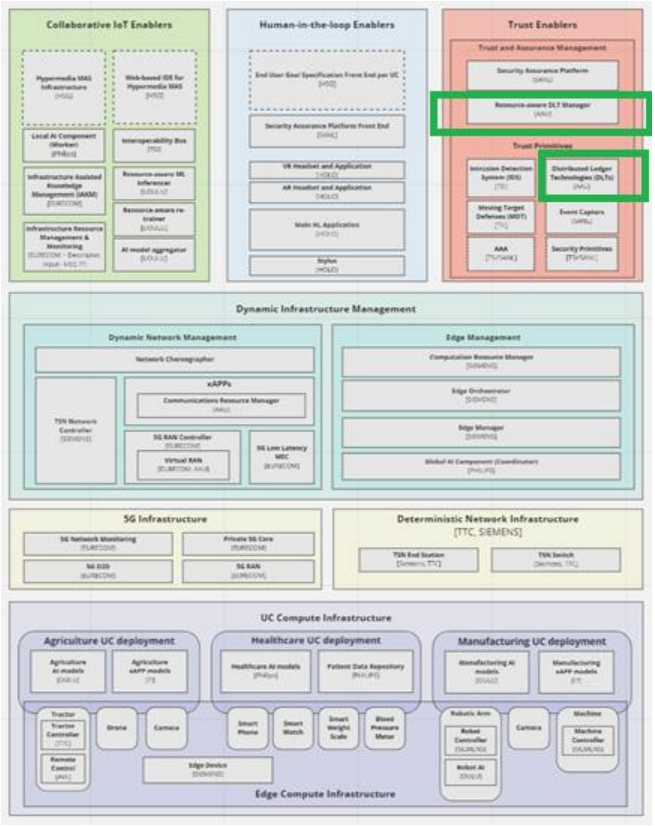


*Figure 1. DLTs as a part of the core trust enablers (top right) of IntellIoT. The green boxes indicate the relevant components for this deliverable. The rest of components in the trustworthiness pillar are addressed in Task 4.4.*

A DLT system offers a tamper-proof ledger that is distributed on a collection of distributed communicating nodes, all sharing the same initial block of information, the genesis block [1]. In order to publish data to the ledger, a node includes data formatted in transactions in a block with a pointer to its previous block, which relates a chain of blocks, the so called DLT. Starting with the adoption of Bitcoin in finance area, DLT has received a lot of attention in the realm of IoT, as the technology promises to help address some of the IoT security and scalability challenges [2].

For instance, in IoT deployments, the recorded data are either centralized or spread out across different heterogeneous parties. These data can be both public or private, which makes it difficult to validate their origin and consistency. In addition, querying and performing operations on the data becomes a challenge due to the incompatibility between different application programming interfaces (APIs). Furthermore, Non-Governmental

Organizations (NGOs), Public and Private sectors, and industrial companies may use different data types and databases, which leads to difficulties when sharing the data [3].

In the scope of IntellIoT project, the DLT provide a transparency, immutability, and distributed platform for distributed IoT application. DLTs solve the problem of trust between participant in a system via mathematical encryption and distributed consensus algorithms. Moreover, with the involvement of smart contracts which is considered as a key innovation of DLTs, this will bring an autonomous decision-making system with wide range of applications and areas.

This task has a two-fold contribution to the IntellIoT framework:

- The **research activities** related to DLT and smart contracts for IoT environments. Specifically, the research questions that have been addressed are: (1) How to support DLT-based communication protocols in resource-constrained wireless IoT networks and what is the trade-off among trust, latency and energy; (2) How to implement DLTs in distributed learning to generate a distributed trust sharing systems and considering resource-constrained IoT system; (3) How to use DLT to do the valuation of data and incentivation for collaborative training and trading of data/models.
- The **implementation** of the **DLT** software components for each of the UCs demos at the end of the project, with each UC having a different level of data privacy.

These efforts contribute directly to Objective 4, which states the following:

**Objective 4:** Enable **security, privacy and trust by design** with **continuous assurance monitoring, assessment and certification** as an integral part of the system, providing **trustworthy integration of third party** IoT devices and services.

Specifically, it contributes to the integration of third-party IoT devices and services through the integration in the architecture of the DLT paradigm. The achievement of this goal is related to the following requirement and KPI:

**Requirement 1: Provide transparent and trusted distributed IoT application based on Distributed Ledger Technologies under the constraints of latency, bandwidth, computing capacities and integrate with third-party IoT services and devices.**

**[KPI-4.2] Delivery of at least 2 DLT implementations that can adjust the level of trust to capabilities of devices, integrate proxies, and conform to certain latency and reliability requirements, such that the level of decentralization of device participation is proportional to its computation-communication capabilities.**

This task will provide 3 DLT implementations, one per UC, considering the communication and computation capabilities of the devices. Besides, we do research on DLT-based communication protocols for distributed IoT networks and the new challenges to the communication network: rather than the conventional uplink-intensive IoT traffic, we have to accommodate a more uplink-downlink balanced IoT traffic. Moreover, each IoT device becomes a DLT client with a digital identity, secured account to generate and exchange transactions in the network. The proposed DLT-based protocols can be deployed under resource-constrained IoT networks. We also address the problem of scalability of centralized IoT networks by using off-chain DLT storage.

## 1.2 Deliverable Outline

The rest of the deliverable is organized as follows: Section 2 explains the background of distributed ledger technologies and the process of selecting the best DLT for IntellIoT. Next, the implementation of DLT & smart contracts for the demonstration of the three UCs is described in section 3, which includes addressing the research problem of resource-constrained devices. Then, section 4 describes the two other research areas relevant for edge collaborative IoT: the implementation in distributed learning scenarios and the valuation of DLT data. The deliverable is concluded in Section 5.

# 2 BACKGROUND

## 2.1 Distributed ledger Technologies

The DLT is a peer-to-peer ledger that records transactions in a network [4]. The transactions represent transfer of data. The transactions are relayed on to the DLTs network, and the peers responsible for validating transactions (also called 'miners') group the transactions together into candidate blocks. The validating peer that successfully solves a complex mathematical puzzle involving the hash of the block, is selected to add its candidate block to the DLTs [5]. Each new added block contains the hash of the previous block. Linking the blocks this way makes the contents of the DLTs immutable, since any changes in a previous block are easily detectable. The remaining nodes verify that the new block is valid and update their copies of the DLTs. Every node carries the same copy of the DLTs, and 'trust-free' distributed networking is established [6].

A high-level overview of the process is shown in Figure 2. The DLT client start creating transactions which are signed with private keys and published to DLT miners. The transactions are here arranged in groups of blocks and broadcasted to all the miner in the DLT network. the initiator of a block or transaction first checks and verifies the difficulty of the block or transaction and then transmits the transaction and block through the gossip protocol. The transmission process requires the initiator to send an inv message to the receiver. The receiver uses the *inv* message to determine whether the block or transaction already exists locally. If the receiver does not store the block or transaction locally, the receiver will send *getdata* information to the initiator. Finally, the initiator sends the block or transaction to the receiver to complete the transmission of the block data. In the process of node interaction, the network latency mainly comes from the difficulty check of the sender and the hash verification of the block as well as the delay in the transmission of *inv* messages, *getdata* messages, and blocks or transactions between the sender and the receiver. In order to optimize the protocol of blocks and reduce the propagation delay, it was proposed in the literature [10–12] that single node optimization and pipelining of the propagation to reduce the propagation delay of DLT networks, respectively.

Among them, single-node optimization stipulates that the sender performs difficulty checking on the block, and the receiver performs hash verification on the block. The streamlining of the propagation process is to pass the block difficulty check and block hash verification to the receiver. Although these optimization schemes reduce the propagation delay to a certain extent, they also bring about other security problems and also affect the performance of the DLT network, and the effect of reducing the total network propagation delay is not obvious. Each miner has its verifier and block to ensure that the real data and the contributions of devices are updated. Each block contains a head and body parts. The blockhead contains a pointer to the next block, and the body part contains a set of validated transaction information. The local models are formed in transaction format and in order to make the solution scalable, the local models are recorded in IFPS storage, such that just a hash version of the data is recorded in the distributed ledger as shown in Figure 1.

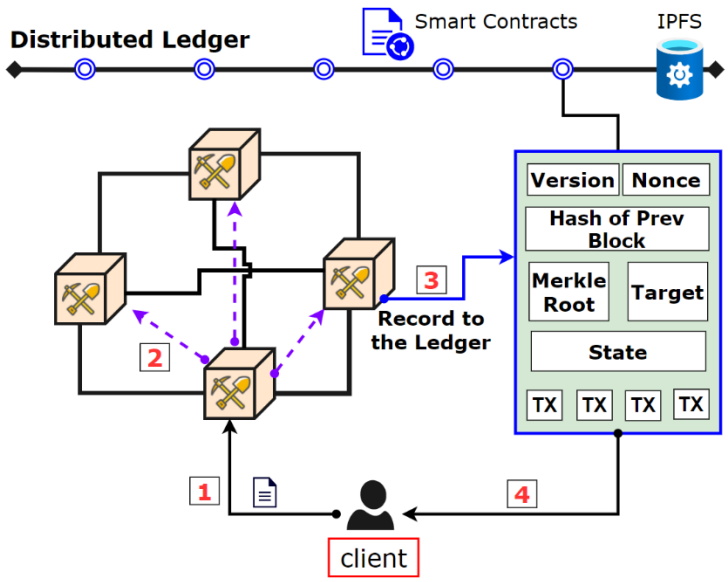


*Figure 2. Distributed Ledger Technology process overview*

## 2.2 Transaction Validation

'Proof-of-Work' is a way of authenticating new transactions into blocks by solving computational puzzles (PoW). The node that adds a new block to the DLTs in PoW is chosen based on a race to discover a nonce that, when hashed with the remainder of the block, generates a result with a specific number of leading zeroes. As a result, PoW-based consensus is computationally demanding and causes delay, both of which are undesirable on the Internet of Things.

'Proof-of-Stake' (PoS) [7] is an alternative consensus mechanism that picks the validating node for adding their candidate block in a pseudo-random manner. The likelihood of a node being picked is proportional to their percentage of the network's stake. PoS has the benefit of not requiring a lot of processing capacity to validate new transactions and add them to the DLTs. PoS, on the other hand, renders the network vulnerable to the 'nothing-at-investment' attack [8, in which nodes may simply establish forks in the chain without having any stake in it or investing a lot of processing resources. Monax and Ethereum's Casper [9], two DLT development platforms that adopt PoS, cope with nothing at stake attacks by reducing the stake of any nodes that produce bogus blocks. Because minimal processing needs are important for resource management, PoS allows us to further investigate the potential of DLTs in IoT.

Further to the above, Smart Contracts bring programmability contracts to the DLTs, in the sense that Smart Contracts execute defined rules autonomously. Smart contracts are deployed in the DLTs with specific addresses, so in order to invoke a function written in a smart contract, transactions are signed off by nodes and addressed to the smart contracts themselves. While smart contracts enforce terms and conditions for transactions in financial applications, they can enforce access control policies in the modular consortium architecture [10].

## 2.3 DLTs Comparison

Although a large number of DLTs are available, the most prominent platforms include Bitcoin, Ethereum, IOTA, and Hyperledger Fabric. In the following, we compare these DLTs in five different aspects: scalability, latency, throughput, security, and the level of smart contract functionalities. Scalability, latency, and throughput are deeply related and of vital importance for IoT applications [11]. For instance, the large number of sensors in smart cities may generate millions of transactions per day. This requires high efficiency of the consensus mechanism, including the way in which transactions are processed by the peers, known as endorsing peers in Hyperledger Fabric and full nodes (peers) in Bitcoin and Ethereum. Regarding latency, the transaction confirmation time must be sufficiently short to avoid queueing in the DLT and to ensure consistency in the ledgers. Bitcoin and Ethereum confirmation times per

transaction are around 10 minutes and 25 seconds, respectively. These latencies might not be suitable for real-time IoT monitoring, while the confirmation time of Fabric and IOTA is much lower [12]. Note that the transaction confirmation *time is only part of end-to-end latency, as it does not account for the communication latency at the radio access network.*

The charge of fees to process the transactions, commonly known as gas is yet another factor to consider selecting the appropriate DLT. These may greatly increase the operational costs of the network, which negatively impacts the throughput of the DLT. On the one hand, transaction fees pose a problem in massive IoT scenarios if the generation of a large number of transactions is essential. On the other hand, these fees may contribute to minimize the number of redundant transactions generated by the sensors, which in turn offloads the DLT. Among the considered DLTs, Ethereum requires fee and gas for each transaction whereas Hyperledger Fabric and IOTA provide free solutions to exchange transactions [13]. It is clear that IoT applications will involve many stakeholders with different roles, functionalities, and information with access rules, identities and security factors.

An important factor to provide security is the support for permissioned and permissionless (i.e., hybrid) solutions to validate participating nodes. Both Ethereum and Hyperledger Fabric support public and private solutions, while Bitcoin and IOTA only provide public ones. Although IoT networks, such as smart cities, may have a large number of stakeholders willing to contribute to the security of a permissionless DLT network, permissioned networks could also be beneficial. For example, in smart homes where the homeowner wants to validate the transactions via home miners or validators [14]. Regarding security, public networks may be more secure than private ones if they are able to provide transparency and distributed storage. For instance, in a permissionless DLT, the data is encrypted and stored in all the devices, which makes it definitely transparent. Besides, the more users a permissionless DLT has, the more secure it is. However, permissionless DLTs are not ideal for enterprise use, where companies deal with highly sensitive data and cannot allow anyone join their network. A permissioned DLT can be altered by its owners, making it more vulnerable to hacking [15]. In addition, permissioned DLTs provide very low or no fee for validation and a faster consensus process.

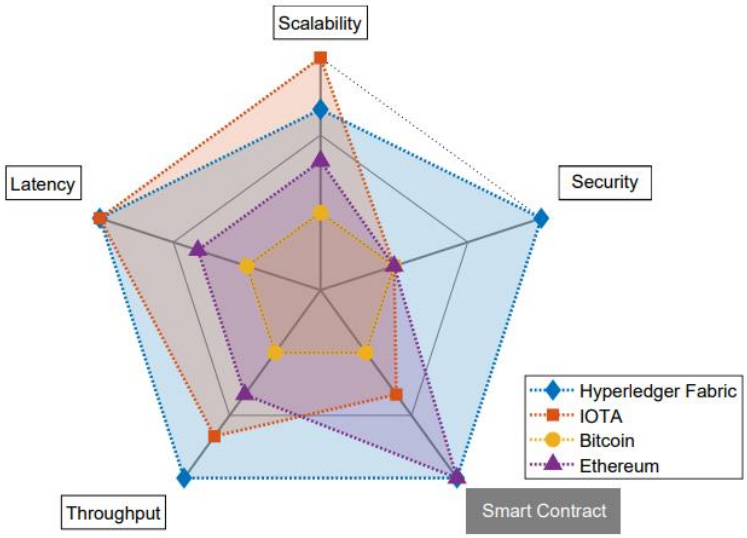


*Figure 3. Comparison of different DLTs*

Finally, smart contracts act as autonomous entities on the ledger that deterministically execute logic expressed as functions of the data that are stored on the ledger. Therefore, smart contracts can be established to have automatic reactions from the DLT network to specific events. For example, in case of carbon emissions, smart contracts can be used for real-time policy enforcement upon changes in the emission patterns. The smart contract feature currently is supported by Ethereum and Hyperledger Fabric (Chaincode) [16]. An IOTA smart contract type is still in progress [17]. Besides, only Hyperledger Fabric supports data confidentially via in-band encryption and guarantees the privacy of data by creating private channels. Hyperledger Fabric provides a solution with various features such as identity

management, transaction integrity, and authorization with a trusted CA. These features are vital in a trusted IoT system.

The comparison of the DLTs mentioned above in these areas is illustrated in Figure 3, where each aspect has been given an abstract score based on the previous discussion. Note that the smart contract aspect is functionality rather than a strict performance indicator and can only be scored qualitatively. This makes it different to the rest of the aspects reflected in the figure, hence, it is shown in a grey background. In this IntellIoT project, because of requirements for implementation in cross-domain application, we have chosen the **Ethereum platform** to implement the system.

# 3 DLT-BASED COMMUNICATION PROTOCOLS AND STORAGE

In this section, we introduce the DLT-based communication in resource-constrained IoT network and distributed storage solution for scalable networks.

## 3.1 DLT-based communication protocols in resource-constrained IoT

We assume the scenario whereby two IoT devices are connected to the peer-to-peer DLT network, whose nodes store the smart contract, maintain the DLT and update it with new information blocks. A device can request the DLT network to update the smart contract state, e.g., to attribute some credit to another device. The other devices are informed of this update when they receive a new block containing the updated state. Finally, the recipient device can verify the new state and release a service to the sender.

The above mechanism enables the exchange of credit in absence of mutual trust, since the economic transaction is certified by a third party, i.e., the DLT network, and hardly reversible [18]. The objective is to assure that each device locally observes the same state of the smart contract, such that the key element in the system design is how each device stays synchronized with the most recent version of the DLT.

Two main types of components are foreseen in this scheme as shown in Figure 4, where Si, Bi act as DLT clients and communicate with DLT manager via wireless base station. In specific:

- **DLT lightweight client**: A light client or light node is a piece of software that connects to full nodes to interact with the DLT. Unlike their full node counterparts, light nodes do not need to run 24/7 or read and write a lot of information on the DLT. In fact, light clients do not interact directly with the DLT; they instead use full nodes as intermediaries. Light clients rely on full nodes for many operations, from requesting the latest headers to asking for the balance of an account.
- **DLT manager**: DLT managers are nodes which have powerful capabilities to do mining process for verifying and have a copy version of entire distributed ledger. When a new set of transactions, packaged in a block, then is broadcast to the entire network by one of the nodes, every other node must verify its validity. DLT consensus is achieved when that block is added to all copies of the DLT. Note that, the DLT lightweight clients typically only download just enough DLT data to process and verify new transactions, and so their computational workload is minimal. Also note that full DLT generally represent large quantities of data.

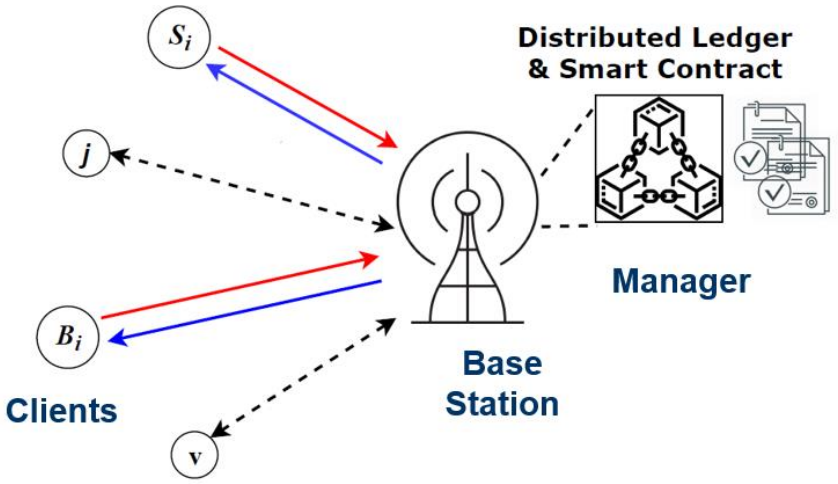


*Figure 4. General architecture of DLT-based IoT networks*

The communication between DLT clients and DLT manager can be categorized into three protocols:

1. **Protocol 1**: The DLT clients, such as Si and Bi in Figure 4, are assumed to have enough capacity to run a full copy of distributed ledger and can accomplish the synchronization process with full on-chain information. However, in realistic, the DLT client is implemented in resource-constraint devices and not capable for running a full node.
2. **Protocol 2**: The DLT client Si and Bi are resource constrained devices which are not capable to run a DLT full ledger, so it just can synchronize a part of data from the distributed ledger, for example, just synchronize the header of blocks instead of synchronizing all data. Via headers of blocks, the client devices can still query the data from the ledger and guarantee about the transparent and availability of the data.
3. **Protocol 3**: In this protocol, the IoT devices will not hold any fragment of data from the ledger, it only uses the open provided interface and send the raw data to the ledger.

The communication workflow between DLT clients and DLT manager is described in the Figure 5, to discuss the protocol overhead imposed by DLTs. A general message exchange in lightweight protocols is depicted in Fig. 5 below shown; note that some specific protocols implementations, e.g. Bitcoin, Ethereum, Fabric, and IOTA, skip some of the phases.

- For instance, phase 0 exists only in Fabric, serving to prepare the transaction by collecting the "endorsement", i.e., authentication, from nodes of the DLT network via the GWs.
- In phase 1, the IoT device invests energy to solve the PoW (this is only encountered in IOTA for preventing spam attack) and digitally signs the transaction (this happens in all the protocols)
- Then, in phase 2, it sends the transaction to the GW. The transaction is forwarded to the DLT network in phase 3, and included in the ledger at the end of phase 4 (this delay exists in blockchains but not in IOTA, where the PoW was done already in phase 1).
- In phase 5, the GW is informed of the validation of the transaction, subsequently informing the IoT device in phases 6 and 7. For instance, in Ethereum-like blockchains, the message in phase 6 is a receipt and the one in phase 7 contains block headers and the proof of inclusion [19]. In incentive-based protocols, step 6 may not be present, and the message in phase 7 may just contain an acknowledgment from a trusted GW. Observe that

with incentive-based protocols, the amount of DLT metadata sent to the device is minimal, because it is filtered by the GWs.

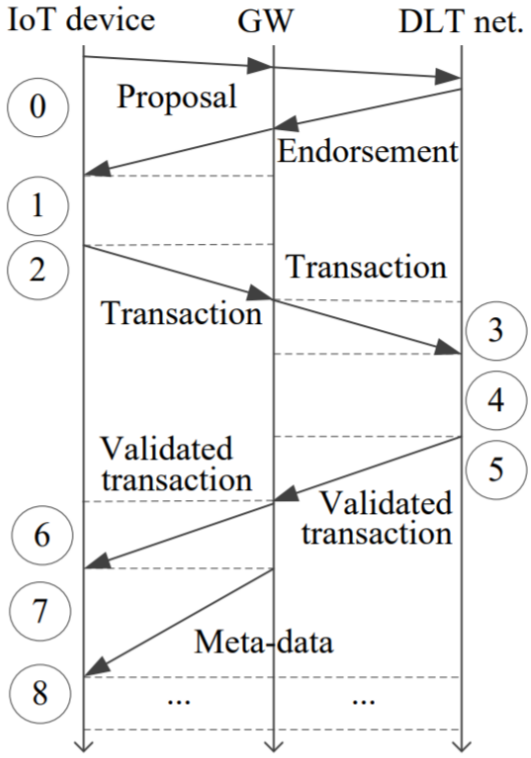


*Figure 5. DLT-based IoT communication protocol [29]*

## 3.2 Distributed Data Storage

In addition to the processing and communication resource challenges, there are also data storage related challenges that must be addressed in the context of DLTs for IoT applications. More details are provided below.

**Scalability Problem**: Large files cannot be efficiently stored on DLTs. On one hand side, the DLT becomes bloated with data that has to be propagated within the DLT network. On the other hand, since the DLT is replicated on many nodes, a lot of storage space is required without serving an immediate purpose, especially if the node operator does not need to view every file that is stored on the DLT. It furthermore leads to an increase in the price of operating DLT nodes because more data needs to be processed, transferred, and stored.

**IPFS (InterPlanetary File System)** is a distributed file system that connects all computing devices to the same file system via peer-to-peer connections. Because the maximum amount of data that can be stored in a DLT block is around 1 MB, storing a big amount of data is problematic. In this project, we integrate IPFS with the whole system as a file system that permits just the hash value returned by IPFS to be kept in a smart contract, allowing a portion of the historical data to be uploaded.

**Implementation**: To develop a Distributed Application (DApp) [30] which are applications that runs on a distributed P2P network rather than a single entity in a local environment more easily, a DLT network was built using Geth [22], an Ethereum RPC client that can be installed and used locally, and Truffle [20] is used as a framework for compiling and distributing smart contracts. The DApp was developed based on a web browser and a web server was built with Node.js. The DApp accesses the smart contract by communicating with the Ethereum network via JSON RPC through

Web3.js [23], an Ethereum JavaScript API. In addition, the web server communicates with the IPFS network to upload data, and to access IPFS, go-ipfs [21], which can run the IPFS network in the local environment, was used. We used the free open source [21] that deals with the web browser-based DApp development tutorial using IPFS for the layout and basic implementation of a web browser-based DApp, and we implemented the system's four recording procedures to finish the system. For example, in Figure 6, we introduce an example of using IPFS for data storage in large and dense network.

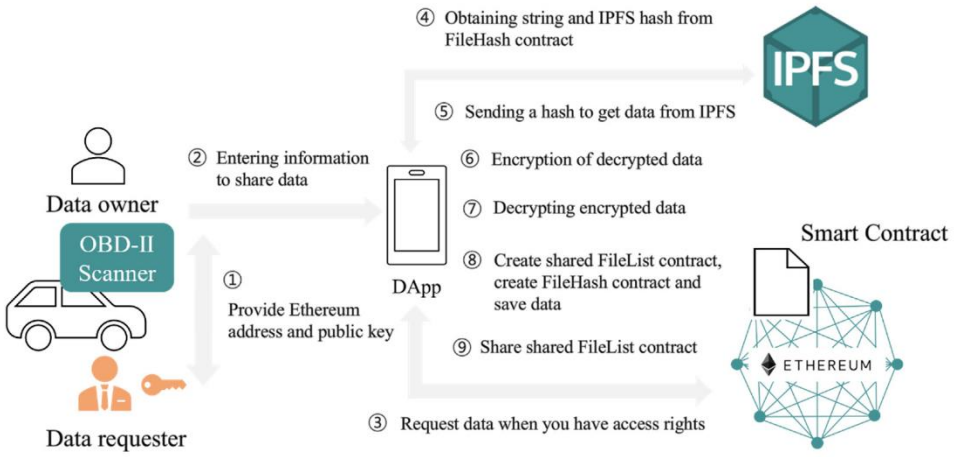


*Figure 6. Example of Off-chain storage solution for DLT-based IoT networks [28]*

## 3.3 IntelIoT's different DLT Implementation Perspectives

Below we provide details on the application of DLTs in the context of the project's different use cases and the framework itself.

### 3.3.1 DLT FOR AGRICULTURE

Agricultural information sharing refers to farm-related activities and administrative data across value chains with the goal of getting the correct data and information to the right person or devices at the right time. Because of the benefits, trusted data sharing in agriculture is essential for the full realization of electronic records. It has the great potential to improve food security while reducing post-harvest losses, facilitating trusted communication and data sharing among critical farm ecosystem participants to enable farm-level decision making, supply and demand visibility, and efficient, trusted, data-driven agriculture.

We carried out an in-depth investigation of the challenges of farming (e.g., tractor management) by conducting design thinking sessions with stakeholders in the ecosystem. Fragmented processes conveyed high transaction cost, complex, and wireless network communication among the participants of the small-scale farming ecosystem makes it very difficult to undertake agribusiness transactions efficiently and transparently across the value chain.

In the standard system, there are some drawbacks.

- First, the lack of information makes it difficult to analyse farming's development and productivity. This would especially be important in developing countries, where food security is closely linked to small-scale farming's performance. In this area, there are three options for modernization and digitalization that can considerably improve the quicker movement of commodities, information, and money [10].
- Second, traceability is currently quite restricted in agricultural ecosystems, and introducing end-to-end visibility across all parts of the agriculture ecosystems can be crucial in assuring food safety and provenance.

- Third, monetary transactions such as trade, credit, and insurance may considerably benefit from access to data from the previous two, making these vital business operations more automated, data-driven, and paperless.

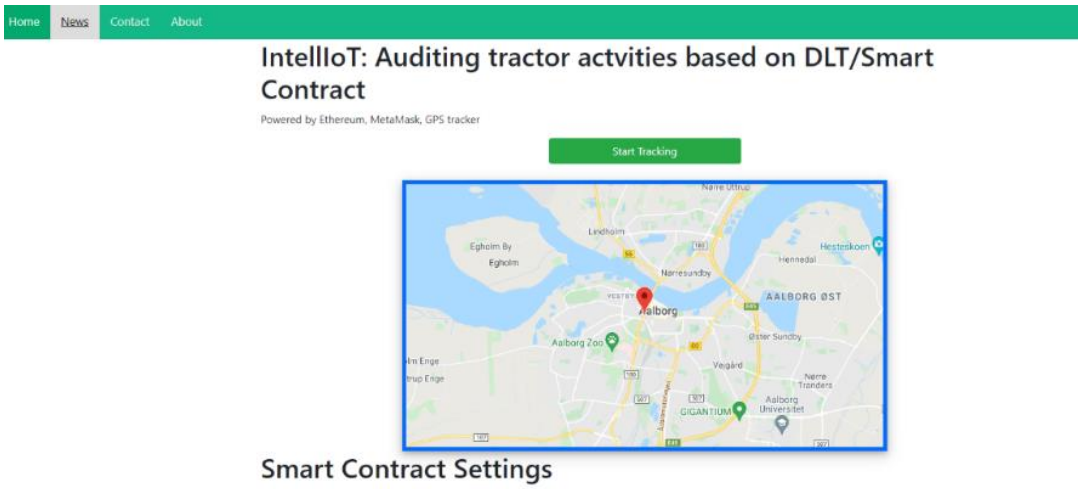


*Figure 7. DLT-enabled Trust location tracking*

In the scope of IntellIoT, initial DLT implementations and smart contracts for recording the location GPS trajectory of the tractor will be implemented in the Tractor Controller. The GPS sensor of the tractor will record the location data via satellites and then format the raw GPS data into DLT transactions.

A lightweight version of a DLT client is setup in the tractor and communicating via secured link with the DLT manager, which is installed in the IoT infrastructure. The DLT client then signs the transaction which includes the GPS data and uploads to the distributed ledger via smart contract function to record the data. The data then will be verified and synchronized among the nodes and server in the DLT network. The GPS data will be transparent and immutable for the administrators as well as customers who rent the tractor. Each tractor owner and customer will be provided a DLT digital identity for the management and control of the tractor.

The smart contracts provide an autonomous strategy for accounting and operating the network. In the agriculture use case, a Smart Contract can manage the tractors activities, location and other statistic metrics. An example of a code snippet for a smart contract to register a tractor with the DLT system is shown below:

```
pragma solidity ^0.5.0;
import "assets.sol";

contract TractorTracker {
    string id;

    function setId(string serial) public {
        id = serial;
    }

    function getId() public constant returns (string) {
        return id;
    }
}
```

Besides, we provide a DLT-based management system for the tractor, where users can manage the tractor via GUI, which can interact with DLT network. Figure 7 shows a proof of concept for the trusted DLT-based GPS tracking.

### 3.3.2 DLT FOR HEALTHCARE

Medical data comprises large files such as medical photographs of patients and films of diagnostic procedures, as well as smaller data items such as text and digital files. As data can only be added to a Blockchain and not removed from it, it results in a DLT ledger's constant extension, which leads to the chain's storage pressure being very high [25]. Furthermore, the DLT system necessitates the local storage of a full ledger by each node. Hence, data redundancy is introduced into the system as a result of recurrent data storage, and increased storage space requirements are imposed for newly added nodes.

In the IntellIoT project, we are addressing two problems: i) security and privacy of data storage and iI) the scalability of data storage in a large network. First, in order to address the security and privacy of data in a healthcare scenario, we integrate the DLT system with the Security Assurance Platform and MTD to detect the malicious sources and record the filtered logs to the DLTs to make sure the data is recorded precisely and correctly. An example of recording medical data using a smart contract is described in the snippet below:

```
pragma solidity ^0.5.5;
/* ABIEncoderV2 is needed to return an array from a function. */
pragma experimental ABIEncoderV2;

contract HeathCareContract {

  /* Every patient or user has a list of medical records. */
  mapping (address => string[]) public medicalRecords;

  function addDocument(string memory documentHash) public returns (uint) {
    address from = msg.sender;
    // push returns the array length
    return medicalRecords[from].push(documentHash) - 1;
  }

  /**
   * Get all medical records of a user.
   */
  function getmedicalRecords(address user) public view returns (string[] memory) {
    return medicalRecords[user];
  }
  mapping (address => address[]) public doctorsPermissions;

  /**
   * Allow a doctor to view all your medicalRecords.
   */
  function giveAccessToDoctor(address doctor) public {
    doctorsPermissions[doctor].push(msg.sender);
  }

  /**
   * Revoke a doctors access to your medicalRecords.
   */
  function revokeAccessFromDoctor(address doctor, uint index) public {
    require(doctorsPermissions[doctor][index] == msg.sender, 'revoke the medical Records of patient.');
    delete doctorsPermissions[doctor][index];
  }
```

Second, in order to address the data storage problem, we deploy an off-chain storage solution. The IPFS system will automatically slice the data and store it in IPFS nodes around the world. Finally, after the hash value of each slice is spliced together, the unique hash address corresponding to the file is calculated. At the same time, IPFS uses content-based addressing instead of traditional address-based addressing. In other words, if the contents of the transmitted file are the same, the corresponding IPFS hash value will not change due to different local storage locations, thus solving the problem of data redundancy in the network caused by repeated data storage. IPFS is therefore an ideal platform for storing bulk medical data.

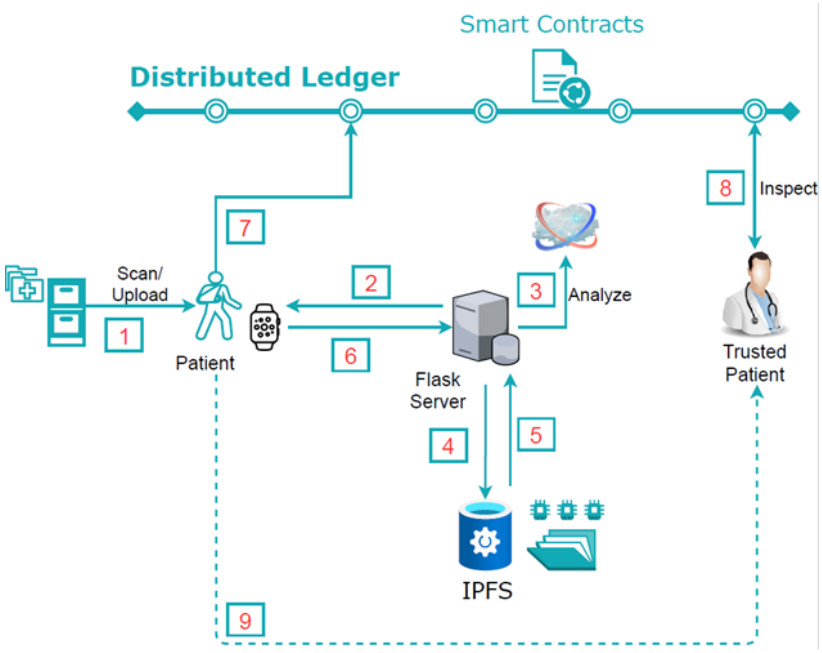


*Figure 8. The application of DLT in trusted medical record*

Figure 8 shows an example of how integrate DLT in healthcare system. The urge for innovation is insatiable when it comes to electronic medical records and data. Hospitals continue to employ age-old data management systems for patient data, and the way patient health information are maintained and safeguarded today does not reflect our technical advances in this field over the last decade. This is partially due to rigorous privacy and security restrictions governing medical data, which have hampered the adoption of cutting-edge technology to make medical data management more visible and beneficial for both patients and clinicians. The application depicts the platform via the eyes of four different actors:

- The administrator is in charge of a group of hospitals and has the highest degree of access in the hierarchy. On their dashboard, they may add a new organization (hospital) to the conglomerate and assign/de-assign hospital administrators.
- The organization (hospital) administrator is in charge of a specific hospital within the conglomerate/solution. They can add new users with the role of patient or doctor, as well as remove existing members.
- The doctor is an organization user with the proper position who has the ability to upload docs for their patients as well as download/view papers for whom they have been authorized access.
- The patient is an organization user with the proper role who may upload documents, examine them, check the document access records, and manage access to their documents on their own. The medical records of patients can be stored in the distributed ledger and available for permissioned actors, such as doctors or hospital. The demo application developed in this context is shown in Figure 9.

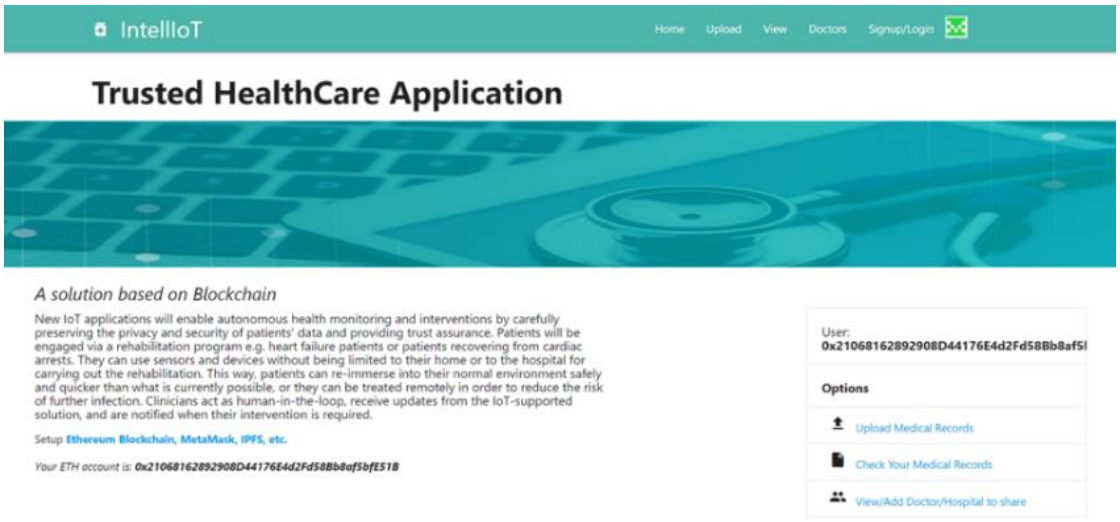


*Figure 9. GUI of the developed application.*

DLT technology presents numerous opportunities for health care; however, it is not fully mature today nor a panacea that can be immediately applied. Several technical, organizational, and behavioral economics challenges must be addressed before a health care DLT can be adopted by organizations nationwide.

### 3.3.3 DLT FOR MANUFACTURING

DLT has a lot of potential for the manufacturing industry. DLT can enable a completely new manufacturing business model by increasing visibility across all elements of the process, from suppliers, strategic sourcing, procurement, and supplier quality through shop floor operations, which include machine-level monitoring and servicing. All industrial organizations rely on supply chains, and the majority of them may benefit from DLT's distributed ledger structure and block-based approach to aggregating value-exchange transactions to boost efficiency. Manufacturers will be able to better meet delivery dates, improve product quality, and ultimately sell more by expanding supplier order accuracy, product quality, and track-and-traceability.

The general architecture of DLT-based manufacturing includes: DLT network which provide a distributed ledger for verification and recording data, physical entities e.g., machines, robots,

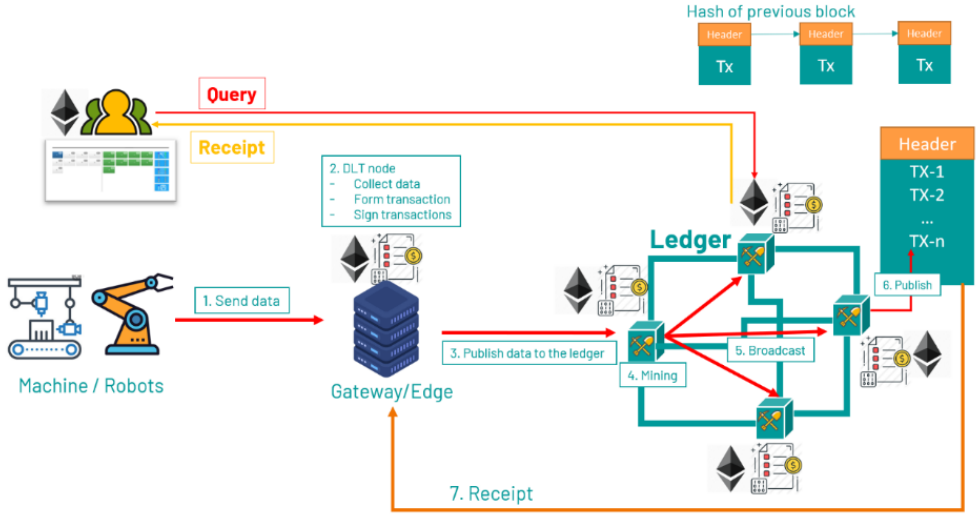


*Figure 10. General Architecture of DLTs in Manufacturing UC*

Figure 10 shows the workflow of the overall DLT-based system in the manufacturing scenario. First, the machines and robots act as DLT clients reporting the operation data in form of DLT transactions via edge devices. The DLT managers are set up in powerful edge nodes. The data then is published to the distributed ledger via smart contracts. Smart contracts are responsible for accepting the published data and autonomously executing the agreements between actors, for example, the machine rental agreements between plant operators and customers. The operation logs of machines are recorded transparently in the distributed ledger, so that it is available for all the actors, and it make convenience for accounting and auditing process. In the scope of this research, we introduce a new integration framework between Ethereum Blockchain and the Universal Robot 5 (UR5) in Fig. 11. The Ethereum DLT with its smart contracts interacts with UR5 via interfaces for accounting the activities and agreements between actors.

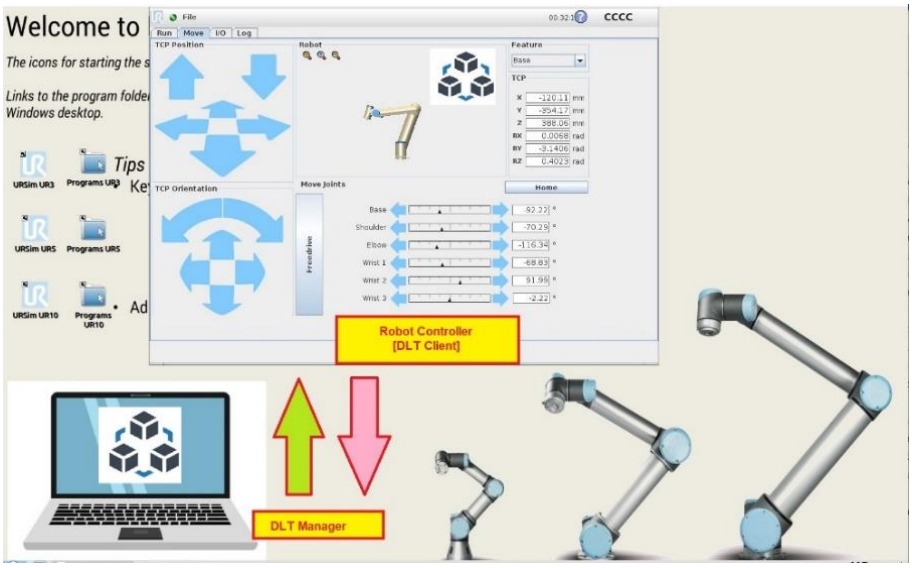


*Figure 11. Proof of concept implementation of shared manufacturing*

To interact with the UR5 robot, we implemented the UR-RTDE [31] library, which allows to query and command the robot via an open interface. Through RTDE, the user can control and manage the robot arm. The smart contract is designed to communicate with the UR5 robot through RTDE, and account the robot activities, for example, working time and movement positions. In a real scenario, it opens the opportunity of pay-per-use schemes in a manufacturing plant between multiple participating organizations. As shown in the snippet below, the RTDE is designed based on Python.

```
#parameters
parser = argparse.ArgumentParser()
parser.add_argument('--host', default='10.0.2.4', help='name of host to connect to (robot controller)')
parser.add_argument('--port', type=int, default=30004, help='port number (30004)')
parser.add_argument('--samples', type=int, default=0, help='number of samples to record')
parser.add_argument('--frequency', type=int, default=125, help='the sampling frequency in Herz')
parser.add_argument('--config', default='record_configuration.xml', help='data configuration file to use (record_configuration.xml)')
parser.add_argument('--output', default='robot_data.csv', help='data output file to write to (robot_data.csv)')
parser.add_argument("--verbose", help="increase output verbosity", action="store_true")
parser.add_argument("--buffered", help="Use buffered receive which doesn't skip data", action="store_true")
parser.add_argument("--binary", help="save the data in binary format", action="store_true")
args = parser.parse_args()

logging.basicConfig(level=logging.INFO)

if args.verbose:
    logging.basicConfig(level=logging.INFO)

conf = rtde_config.ConfigFile(args.config)
output_names, output_types = conf.get_recipe('out')

con = rtde.RTDE(args.host, args.port)
con.connect()

# get controller version
con.get_controller_version()

# setup recipes
if not con.send_output_setup(output_names, output_types, frequency = args.frequency):
    logging.error('Unable to configure output')
    sys.exit()

#start data synchronization
if not con.send_start():
    logging.error('Unable to start synchronization')
    sys.exit()
```

### 3.3.4 DLT IN INTELLIOT'S TRUSTWORTHINESS PILLAR

While DLTs and associated enablers are developed within Task 3.4, there is also a strong link and associated work being carried out in the context of Task 4.4 ("Trustworthy infrastructure by design"), as DLTs are a core part of the trust-by-design approach within the IntellIoT framework.

More specifically, these joint efforts pertain to the integration of the DLT enablers with the rest of the Trust Enablers of IntellIoT which are being developed in Task 4.4, including the Security Assurance Platform (SAP), the Trust Intrusion Detection System (Trust IDS), the Moving Target Defences (MTDs), and the multi-layer monitoring through Event Captors and the EVEREST Monitoring Engine of SAP. For a full list of IntellIoT's Trust Enablers, their specification, and their interplay, we refer the reader to deliverable D2.3 – "High level architecture (first version)", where all of these have been extensively documented.

This integration between the Trust Enablers, mainly taking place through direct integration between the SAP and the DLT Manager, facilitates the trust-by-design approach of IntellIoT, as it allows the recording of security and privacy - pertinent evidence from said Trust Enablers, via SAP, in the Ledger, in a tamper-proof manner. This provides an additional layer of trust for said evidence aggregated at the SAP, which may include:

1) Evidence generated internally at the SAP (e.g., vulnerability or dynamic testing assessment results).
2) Evidence the SAP collects from monitoring and interacting with other Trust Enablers (e.g., Trust IDS alerts, or triggered MTD strategies)
3) Monitoring Evidence from monitoring the Event Captors deployed across the various layers of IntellIoT and the protected deployment.

Thus, through interaction between the SAP and the DLT Manager, the above types of Evidence are recorded in the Ledger in an automated manner, and the entries (more specifically, transaction and block IDs) are returned to the SAP, to be provided to the SAP operator for verification (e.g., in the case of any audit). Additional details on this integration,

and how each of the above three cases are relayed from the SAP to the DLT Manager, are provided in deliverable D4.4 – "Trust mechanisms (first version)" of Task 4.4. Said deliverable also includes details on the design and development of the rest of the Trust Enablers highlighted above.

# 4 DLT ASSISTED DISTRIBUTED LEARNING

In this section, we introduce the general framework which integrates the DLTs with Federated Learning to improve the security and privacy of Federated Learning in wireless IoT network. In addition, we also study deeply about the value of data generated from IoT devices which usually not considered significantly for the use.

## 4.1 Overview

Federated Learning (FL) is a machine learning scheme where multiple entities (clients) collaborate via a central server in training a model without sharing their available raw data [27]. Instead, clients perform computations on their data and transfer obtained local learning parameters updates to the server for the aggregation process. The aggregated model, i.e., the global model, is broadcast back to the clients for the next round of local computations resulting in the local learning parameters. The interaction between the server and clients to solve the learning problem continues until an acceptable level of model accuracy is achieved [5]. The overall process is shown in Figure 12.

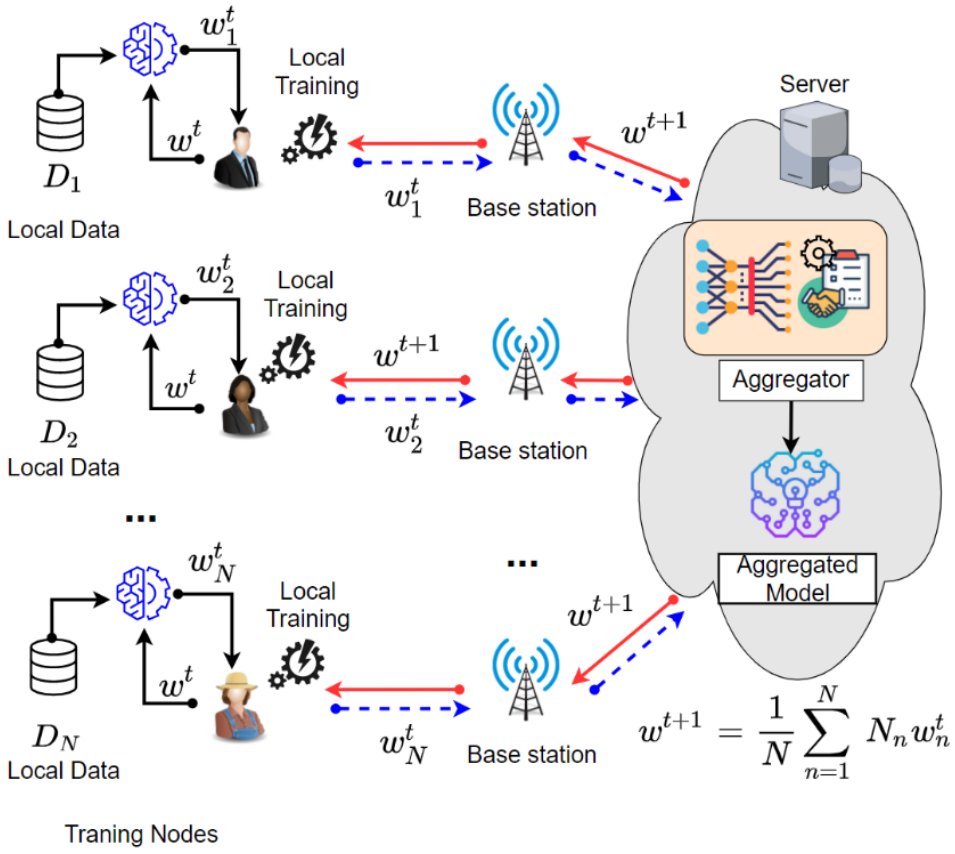


*Figure 12. Typical Federated Learning scheme*

In this manner, FL offers (i) privacy-preserving benefits in the model training approach by not requiring clients to share their local data to the server, and consequently, (ii) lower communication overhead by offering a distributed model training paradigm and exchange of model parameters only. Therefore, FL can serve as an enabling technology for ML model training at edge networks.

However, today's systems, for example, a common hospital IT environment, are often centralized and lack data integrity, trust, security, and transparency. Traditional data markets are often implemented as a centralized service platform that collects data from data owners and sells raw or processed data to the consumers. This approach leaves the platform as a single point of security vulnerability for the data market, and corruption on the platform servers may

lead to severe security issues including leakage of private data, faulty computation results, and manipulation of data price.

In addition, the data can be both public or private, which makes it difficult to validate their origin and consistency. Besides, querying and performing operations on the data becomes a challenge due to the incompatibility between different application programming interfaces (APIs) among organizations that may use different data types and databases, which leads to difficulties when sharing the data. On the other hand, traditional exchange systems (e.g., Paypal or Ebay) are vulnerable to a single point of failure, the lack of trust, transparency, and incentive for data trading, which is preventing the availability of digital information from data providers to customers.

## 4.2 DLT-based infrastructure for Federated Learning

In DLT-based Federated Learning, each learning device trains its own model with its own local data set to guarantee privacy. Each device is connected virtually to a miner in the DLT, i.e., the connection between the miner and device is variable and we assumed that they are connected automatically. This strategy ensures constant communication between the device and the DLT layer, even if the miner fails. The miner connected to the device acts as the leader of the device and is responsible for uploading and downloading data of the device's data. In each iteration, the device receives the latest global model stored in the DLT system and trains a new round of local models using local data. After completing local training, the device uploads the local model to the adjacent miner and starts a global aggregation process. The process allows all devices to download the latest information from adjacent miners to receive the devices' scores and global model updates. More details on this process and the device scores will be described later in this section. Finally, each device uploads its local model and scores and proceeds to a new iteration

The training process is repeated until the global model has achieved satisfactory accuracy and convergence. In the DLT layer, duplicated versions of distributed ledger are kept by all miners. The DLT miners are members of specific DLT networks and actively participate in the trading process. They validate transactions through their own distributed ledger and store the scores of the models and devices in FL. The miners can be personal computers, enterprise servers, or cloud-based nodes if they voluntarily download the mining software. Each miner has its verifier and block to ensure that the real models and the devices' scores are updated. Each block contains a head and body parts. The blockhead contains a pointer to the next block, and the body part contains a set of validated transaction information. The local models are formed in transaction format and to address scalability issues, the local models are recorded in IFPS storage, and just hash versions of models are recorded in the distributed ledger.

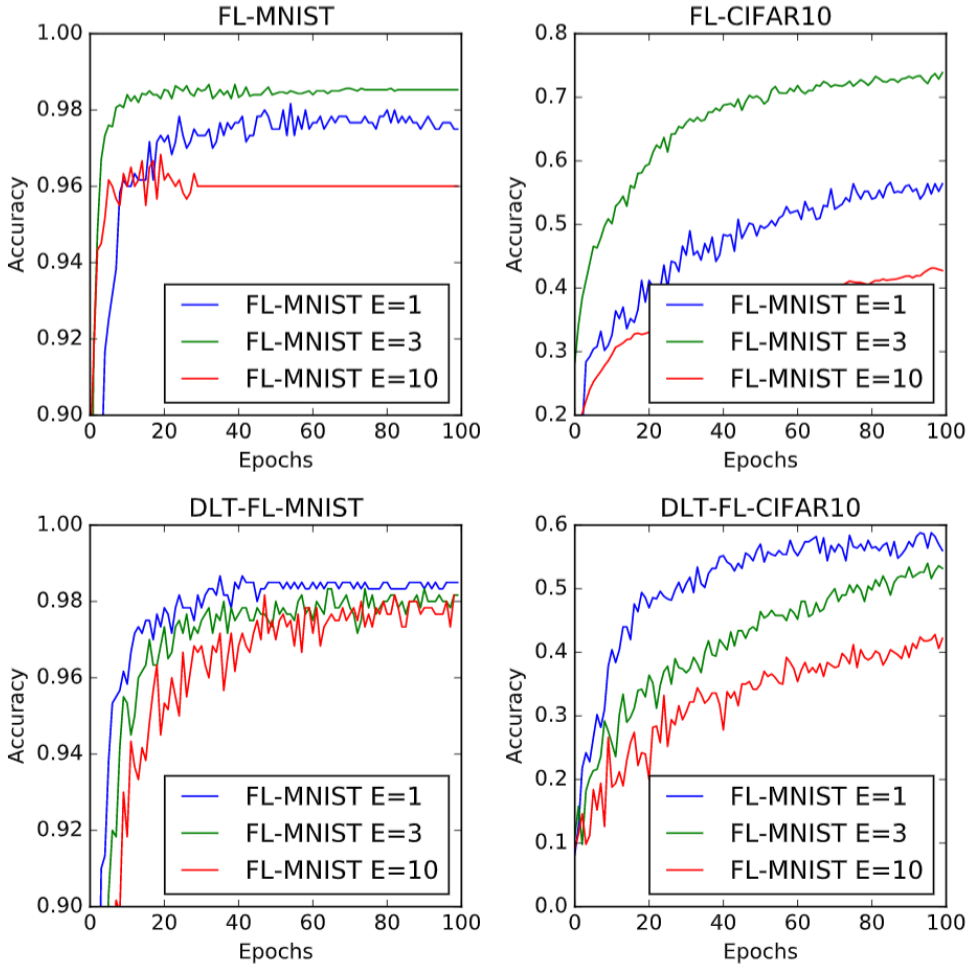


*Figure 13. Performance comparison of standard FL and DLT-based FL*

The basic comparison between standard FL and DLT-based FL is presented in Fig.13. The accuracy is similar in both standard FL and DLT-based FL, but the time required for convergence of DLT-based FL is higher than standard FL because of extra verification and consensus in the system

## 4.3 Freshness and valuation of data

In 2025, the volume of sensing data generated by personal IoT devices is expected to reach 79.4 ZB globally. Many attempts have been made to improve and adapt business workflows to exploit the availability of IoT data [2], for example, using IoT data for training machine learning models and sharing data are the most popular approaches. Various services for using and sharing of IoT data are emerging, connecting various devices and distributed IoT data sources, thereby facilitating data providers to exchange their data

For example, in the medical surveillance, vital gathered data of patients, e.g., heartbeat, x-ray that could be used for learning diagnostic support systems for detect diseases might be in possession of different hospitals, each of which have different data sources, for instance, from a specific geographical region with different demographics , the data needed to train good models often exists but is not easy to leverage as it is distributed and owned by multiple involved parties. Therefore, by combining the available data from various data silos, the hospitals could for their medical applications using their own data. As hospitals would benefit from better machine learning models obtained through data sharing there is an incentive for patients to give access to their personal data and a need for a distributed ML model marketplace.

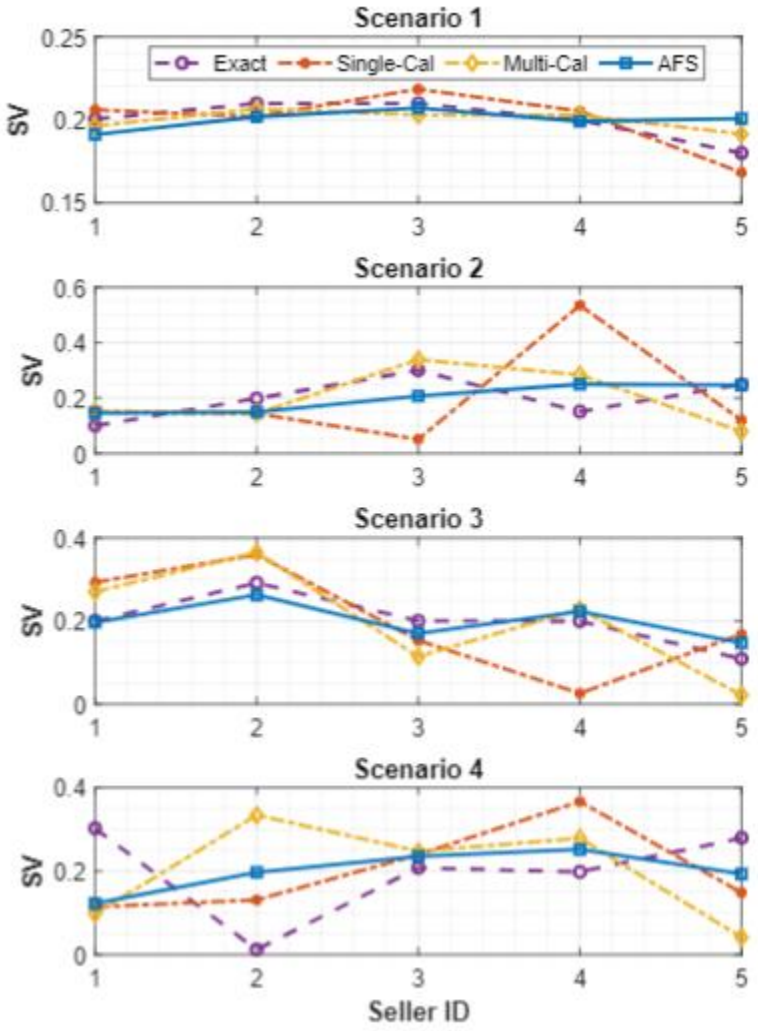


*Figure 14: Valuation of data with different Algorithms*

A conventional data sharing is often deployed as a centralized service platform that gathers and sells raw or processed data from data owners (e.g., the trained learning models) to the consumers [12] [13]. This leads to two important concerns. First, this strategy exposes the platform as a single point of security risk; the malfunctioning platform servers has serious security concerns including data leakage, inaccurate calculations results, and manipulation of data price. Second, collaboration for model training raises questions in terms of how to motivate participants to participate in such a training endeavour. The incentive for each IoT client based on their contribution should be fair and transparent. These features are not present in a standard FL setup, as in many applications there is no clear and natural incentive mechanism for involved participants to provide quality information. This calls for a carefully designed mechanisms to reward parties economically and thus incentivize participation [14], [15]. For example, a fixed price per data point could motivate participants to collect massive amounts of low quality or fake data if there is no intermediary process to check quality of training data. Besides, another reason that may disincentives parties from sharing data could stem from privacy and integrity concerns regarding the use of participant's data once it is shared. For instance, the sellers can re-use the data which has already been sold.

With the aforementioned motivation, we propose a Blockchain-based model trading system which enables a secured and trusted marketplace to collaboratively train models as well as guarantees fair incentives for every participant and privacy of data. Based on the quality of the uploaded models, which is quantified by using a distributed Data Shapley

Value (DSV), the participants can get the incentive based on the updated models, for example, as tokens or fiats. Note that based on our proposed system, the parties do not need to share their local data, but only provide customized models or query interface to the marketplace. Consequently, the proposed system allows multiple participants to jointly train the ML models on the marketplace based on their own training data. Buyers who need to train their ML model will pay to the market for the improvement of their model, and sellers who sell their contribution to train the ML models will get paid by the market via smart contracts. In Figure 14, we show the contribution of different involved participants calculated based on Shaley Value.

# 5 CONCLUSION & FUTURE DIRECTION

In this first phase of the project (i.e., Cycle 1), we successfully designed the DLT-based mechanisms for the three use cases healthcare, agriculture, and manufacturing. The DLT provides the transparency and immutability platform for building a trusted system, and with the deployment of smart contracts, it helps to design different type of distributed applications for various domains. In conclusion, we documented the integration and application of Distributed Ledger Technologies (DLTs) to the three domains. The natural characteristics of DLTs can be used for accounting and auditing recorded data. The DLTs guarantee the privacy and security of data in homogenous networks. In addition, the integration of DLTs with HyperMAS in D3.1 and the SAP/MTD in D4.4 could support the system in terms of security and privacy. The operators and analysis records of MTD and HyperMAS are located in the distributed ledger.

The resulting designs and developments as part of IntellIoT have contributed to the following publications:

- Lam Duc Nguyen, Shashi Raj Pandey, Soret Beatriz, Arne Bröring, and Petar Popovski, "A Marketplace for Trading AI Models based on Blockchain and Incentives for IoT Data". IEEE Transaction (2021; *under review*)
- Nguyen, Lam Duc, Israel Leyva-Mayorga, Amari N. Lewis, and Petar Popovski. "Modeling and analysis of data trading on blockchain-based market in iot networks." *IEEE Internet of Things Journal* 8, no. 8 (2021): 6487-6497.
- Nguyen, Lam Duc, Amari N. Lewis, Israel Leyva-Mayorga, Amelia Regan, and Petar Popovski. "B-ETS: A Trusted Blockchain-based Emissions Trading System for Vehicle-to-Vehicle Networks." VEHITS 2021) - *Best Paper Award*.
- Soret, Beatriz, Lam D. Nguyen, Jan Seeger, Arne Bröring, Chaouki Ben Issaid, Sumudu Samarakoon, Anis El Gabli, Vivek Kulkarni, Mehdi Bennis, and Petar Popovski. "Learning, Computing, and Trustworthiness in Intelligent IoT Environments: Performance-Energy Tradeoffs." IEEE Transaction on Green Communication and Networking (2021; *under review*).

In the Cycle 2 of the project for deliverable D3.8, we will focus on the i) the practical performance of experiments of DLTs in various application domains, ii) the valuation of IoT data in each use cases and the reliability of communication and computation overhead of actors in wireless IoT networks, and ii) we can realize that the in the first cycle only valuation of data generated is considered base on the Shapley value in DLT-based federated learning environment, but the communication overhead of integration DLT and federated learning is not considered yet, so that a study about the connectivity aspect among actors in distributed intelligent environment is needed.

# REFERENCES

[1] Urquhart, Andrew. "The inefficiency of Bitcoin." Economics Letters 148 (2016): 80-82.

[2] L. D. Nguyen, I. Leyva-Mayorga, A. N. Lewis and P. Popovski, "Modeling and Analysis of Data Trading on DLT-Based Market in IoT Networks," in *IEEE Internet of Things Journal*, vol. 8, no. 8, pp. 6487-6497, 15 April15, 2021, doi: 10.1109/JIOT.2021.3051923.

[3] Fernández-Caramés, Tiago M., and Paula Fraga-Lamas. "A Review on the Use of Blockchain for the Internet of Things." Ieee Access 6 (2018): 32979-33001.

[4] Huh, Seyoung, Sangrae Cho, and Soohyung Kim. "Managing IoT devices using blockchain platform." In 2017 19th international conference on advanced communication technology (ICACT), pp. 464-467. IEEE, 2017.

[5] Wang, Wenbo, Dinh Thai Hoang, Peizhao Hu, Zehui Xiong, Dusit Niyato, Ping Wang, Yonggang Wen, and Dong In Kim. "A survey on consensus mechanisms and mining strategy management in blockchain networks." IEEE Access 7 (2019): 22328-22370.

[6] Tschorsch, Florian, and Björn Scheuermann. "Bitcoin and beyond: A technical survey on decentralized digital currencies." IEEE Communications Surveys & Tutorials 18, no. 3 (2016): 2084-2123.

[7] Gaži, Peter, Aggelos Kiayias, and Dionysis Zindros. "Proof-of-stake sidechains." In 2019 IEEE Symposium on Security and Privacy (SP), pp. 139-156. IEEE, 2019.

[8] Gaži, Peter, Aggelos Kiayias, and Alexander Russell. "Stake-bleeding attacks on proof-of-stake blockchains." In 2018 Crypto Valley Conference on Blockchain Technology (CVCBT), pp. 85-92. IEEE, 2018.

[9] Buterin, Vitalik, Daniël Reijsbergen, Stefanos Leonardos, and Georgios Piliouras. "Incentives in Ethereum's hybrid Casper protocol." International Journal of Network Management 30, no. 5 (2020): e2098.

[10] Zhang, Yuanyu, Shoji Kasahara, Yulong Shen, Xiaohong Jiang, and Jianxiong Wan. "Smart contract-based access control for the internet of things." IEEE Internet of Things Journal 6, no. 2 (2018): 1594-1605.

[11] Putra, Dwiyan Rezkia, Bayu Anggorojati, and Ardhi Putra Pratama Hartono. "Blockchain and smart-contract for scalable access control in internet of things." In 2019 International Conference on ICT for Smart Society (ICISS), vol. 7, pp. 1-5. IEEE, 2019.

[12] Onik, Md Mehedi Hassan, and Mahdi H. Miraz. "Performance analytical comparison of blockchain-as-a-service (baas) platforms." In International Conference for Emerging Technologies in Computing, pp. 3-18. Springer, Cham, 2019.

[13] Pustišek, M. and Kos, A., 2018. Approaches to front-end IoT application development for the ethereum blockchain. Procedia Computer Science, 129, pp.410-419.

[14] Dorri, Ali, Salil S. Kanhere, Raja Jurdak, and Praveen Gauravaram. "Blockchain for IoT security and privacy: The case study of a smart home." In 2017 IEEE international conference on pervasive computing and communications workshops (PerCom workshops), pp. 618-623. IEEE, 2017.

[15] Hong, Sunghyuck. "Survey on analysis and countermeasure for hacking attacks to cryptocurrency exchange." Journal of the Korea Convergence Society 10, no. 10 (2019): 1-6.

[16] Androulaki, Elli, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart et al. "Hyperledger fabric: a distributed operating system for permissioned blockchains." In Proceedings of the thirteenth EuroSys conference, pp. 1-15. 2018.

[17] Androulaki, Elli, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart et al. "Hyperledger fabric: a distributed operating system for permissioned blockchains." In Proceedings of the thirteenth EuroSys conference, pp. 1-15. 2018.

[18] Androulaki, Elli, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart et al. "Hyperledger fabric: a distributed operating system for permissioned blockchains." In Proceedings of the thirteenth EuroSys conference, pp. 1-15. 2018.

[19] Peng, Kun. "A general, flexible and efficient proof of inclusion and exclusion." In Cryptographers' Track at the RSA Conference, pp. 33-48. Springer, Berlin, Heidelberg, 2011.

[20] D. -L. Nguyen, I. Leyva-Mayorga and P. Popovski, "Witness-based Approach for Scaling Distributed Ledgers to Massive IoT Scenarios," *2020 IEEE 6th World Forum on Internet of Things (WF-IoT)*, 2020, pp. 1-6, doi: 10.1109/WF-IoT48130.2020.9221269.

[21] L. D. Nguyen, A. E. Kalor, I. Leyva-Mayorga and P. Popovski, "Trusted Wireless Monitoring Based on Distributed Ledgers over NB-IoT Connectivity," in *IEEE Communications Magazine*, vol. 58, no. 6, pp. 77-83, June 2020, doi: 10.1109/MCOM.001.2000116.

[22] https://geth.ethereum.org/

[23] https://nodejs.org/en/

[24] Bermeo-Almeida, O., Cardenas-Rodriguez, M., Samaniego-Cobo, T., Ferruzola-Gómez, E., Cabezas-Cabezas, R. and Bazán-Vera, W., 2018, November. Blockchain in agriculture: A systematic literature review. In International Conference on Technologies and Innovation (pp. 44-56). Springer, Cham.

[25] Hölbl, M., Kompara, M., Kamišalić, A. and Nemec Zlatolas, L., 2018. A systematic review of the use of blockchain in healthcare. Symmetry, 10(10), p.470.

[26] Hölbl, M., Kompara, M., Kamišalić, A. and Nemec Zlatolas, L., 2018. A systematic review of the use of blockchain in healthcare. Symmetry, 10(10), p.470.

[27] Lu, Y., Huang, X., Dai, Y., Maharjan, S. and Zhang, Y., 2019. Blockchain and federated learning for privacy-preserved data sharing in industrial IoT. IEEE Transactions on Industrial Informatics, 16(6), pp.4177-4186.

[28] Ye, H.; Park, S. Reliable Vehicle Data Storage Using Blockchain and IPFS. Electronics 2021, 10, 1130. https://doi.org/10.3390/electronics10101130

[29] Danzi, Pietro, Anders E. Kalor, Rene B. Sorensen, Alexander K. Hagelskjær, Lam D. Nguyen, Cedomir Stefanovic, and Petar Popovski. "Communication aspects of the integration of wireless iot devices with distributed ledger technology." IEEE Network 34, no. 1 (2020): 47-53.

[30] https://ethereum.org/en/dapps/

[31] https://www.universal-robots.com/articles/ur/interface-communication/real-time-data-exchange-rtde-guide/