Lightweight End-to-End Blockchain for IoT Applications

Seungcheol Lee[1], Jaehyun Lee[1], Sengphil Hong[2], and Jae-Hoon Kim[1*]
[1] Dept. of Industrial Engineering, Ajou University
Suwon, S. Korea
[e-mail: {ooo890, leejh324, jayhoon}@ajou.ac.kr]
[2] Vice President, HANCOM WITH
Sungnam, S. Korea
[e-mail: sengphil@hancomwith.com]
*Corresponding author: Jae-Hoon Kim

Received September 14, 2019; revised December 18, 2019; revised March 13, 2020; revised June 9, 2020; accepted July 10, 2020; published August 31, 2020

Abstract

Internet of Things (IoT) networks composed of a large number of sensors and actuators generate a huge volume of data and control commands, which should be enforced by strong data reliability. The end-to-end data reliability of IoT networks is an essential industrial enabler. Blockchain technology can provide strong data reliability and integrity within IoT networks. We designed a lightweight end-to-end blockchain network that applies to common IoT applications. Its enhanced modular architecture and lightweight consensus mechanism guarantee its practical applicability for general IoT applications. In addition, the proposed blockchain network is highly software compatible because it adopts the Hyperledger development environment. Directly embedding the proposed blockchain middleware platform in small computing devices proves its practicability.

Keywords: IoT, Internet of Things, lightweight blockchain, Hyperledger

A preliminary version of this paper appeared in ICCA-KSB, November 16–17, 2018, Seoul, Korea. This version includes a concrete analysis and supporting implementation results for the previous version. This work was supported in part by a grant from the Institute for Information and Communications Technology Promotion (IITP) funded by the Korean Government (Ministry of Science and Information Technology) (Versatile Network System Architecture for Multi-Dimensional Diversity) under Grant 2016000160, and in part by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (Ministry of Science and Information Technology) under Grant 2020R1F1A1049553

http://doi.org/10.3837/tiis.2020.08.004 **ISSN : 1976-7277**

1. Introduction

The huge volume of generated, collected, and utilized data has changed current industrial systems. The data now facilitate industrial transformations and act as key elements that promote the integration of virtual spaces into physical spaces. The blockchain, which can implement transparent and reliable data transactions, has emerged as an innovative growth engine. Blockchain technology applies to the entire industrial sector as the core of a fourth-industrial-revolution enabler. A blockchain's core area applies to traditional industrial areas, e.g., finance, logistics, and commerce.

The blockchain is a general decentralized system that guarantees data reliability in existing industrial domains. In recent years, these domains have expanded to include user-oriented application areas, e.g., payment and personal-information exchanges. A blockchain service is generally developed in the form of distributed applications over a blockchain network.

The Ethereum network is a representative blockchain service [1]. It is more popular among developers than other blockchain networks. It supports Web 3 as a front-end development environment that utilizes both javascript and HTML/CSS. It also supports various development languages, e.g., Solidity and Serpent, to ensure the blockchain's network accessibility. The EOS [2] network has received attention as another representative blockchain network. It has a lower delay for activating distributed applications (dApps) by providing a faster network-processing speed than Ethereum.

Unfortunately, common blockchain networks, e.g., Ethereum or EOS, have an inherent problem in targeting network nodes with considerable computing power. For example, *ethash*, Ethereum's proof-of-work module, contains a large hexadecimal code file, called a directed acyclic graph (DAG), which runs in computer memory. This is a considerable obstacle for an IoT node with very little computing power or memory. Ethereum needs the GPU mining in the consensus process. In initial phase of Ethereum network activation, relatively small amount of RAM is enough for operation. However, the required amount of RAM increases time to time. The incremental RAM requirement restricts the use of Ethereum on IoT devices.

Most field data are generated by Internet of Things (IoT) sensors. A blockchain network in an IoT environment has great significance for end-to-end data integrity. The chain-agent model is suggested to cover the end-to-end data integrity of a blockchain network. **Fig. 1** shows a chain-agent model where an agent node drives a blockchain, and IoT sensor/actuator nodes send the collected data to the agent node. The data integrity belongs to the block generation by the agent nodes.

Since the chain-agent model constitutes a blockchain between agent nodes with high computing power, a certified blockchain network, e.g., Ethereum, EOS, or Hyperledger [3], can be applied. However, the data integrity powered by the blockchain network cannot be applied from the IoT sensor/actuator nodes to the agent nodes. There is no choice but to apply general-purpose communication security between them. This limitation is already known as the *oracle* problem in blockchain networks.

Typical blockchains and dApps cannot directly access data from outside of their network. A dApp often needs access to electronic data from the outside world, e.g., data relevant to a contractual agreement, so they access *oracles*. These oracles are services that send and verify real-world occurrences and submit this data to the dApps or blockchains, triggering state changes on the blockchains.

The main challenge with oracles is that people need to trust these outside data sources, whether they come from a website or a sensor. Since oracles are third-party services that are not part of the blockchain consensus mechanism, they are not subject to the underlying security mechanisms that a blockchain network provides. One could replicate "man-in-the-middle" attacks, standing between the dApps and the oracles. Some trusted computing techniques can be used to address these issues. However, this topic will need more attention, as secure oracles are a bottleneck for data integrity. If oracle security is not adequately provided, it will be a showstopper for widespread blockchain-network implementations.

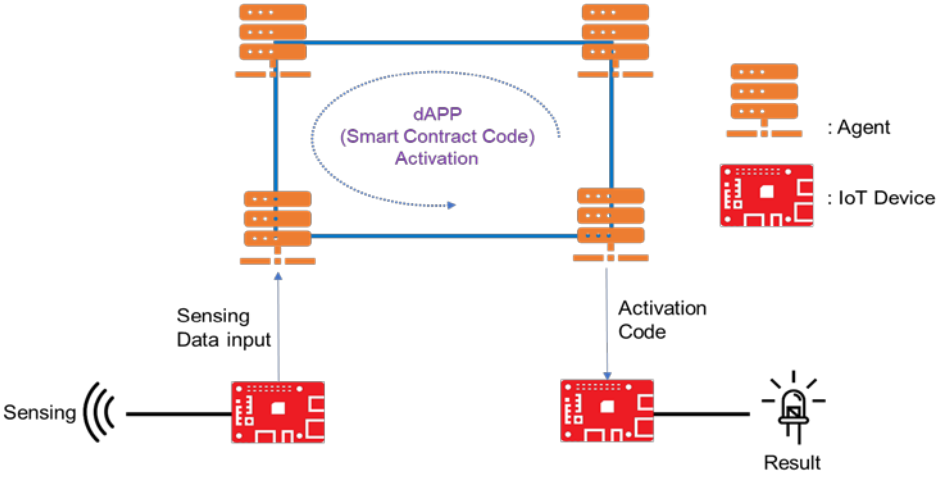


Fig. 1. Chain-Agent Model

An end-to-end blockchain network, as in **Fig. 2**, fundamentally solves the oracle problem. Its structure ensures the data integrity of IoT applications. A small blockchain middleware module is embedded in each sensor device. A dApp uses data gathered from the sensor devices and generates the proper commands to the actuator devices. The embedded blockchain module guarantees the data integrity of the transactions between the devices and the dApp. In addition, an external public/private cloud system extends the use of the dApp to external users. To implement the end-to-end model, A developer can use a lightweight peer-to-peer data-sharing scheme such as a gossip protocol, that considers an IoT device's small computing power and memory resources. A lightweight consensus algorithm and network scaling for data distribution are also required.

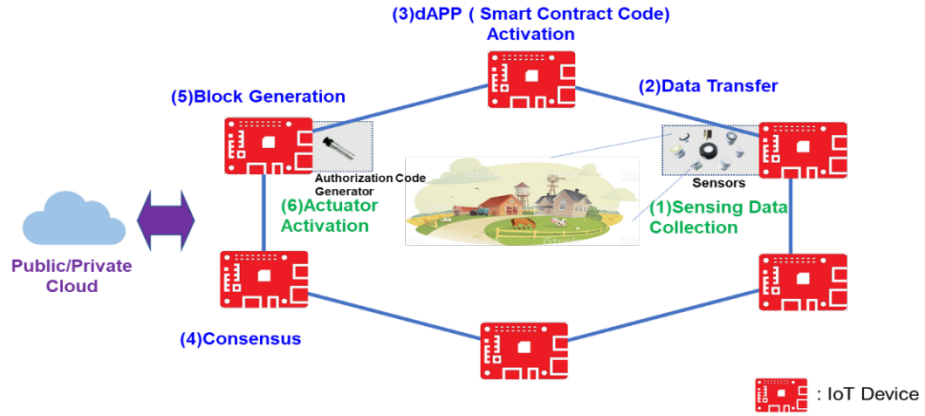


Fig. 2. End-to-End Model

In this article, we propose a practical end-to-end (E2E) blockchain network for IoT applications. A lightweight blockchain middleware module can be easily embedded in popular small devices. Both Advanced RISC Machine (ARM) and Intel architectures are supported by our lightweight blockchain middleware. To show the usefulness of our developed solution, a test dApp is built for a smart farm. A chemical-detecting sensor in the smart farm collects chemical data in real-time. A blockchain middleware module embedded in the sensor devices guarantees a secured data transaction for the dApp. Moreover, the sensor device builds blocks and performs the consensus as a blockchain validator. The proposed lightweight blockchain middleware module can be a technical enabler for practical lightweight end-to-end blockchain solutions for IoT applications.

2. Related Work

IoT devices has promoted huge useful applications such as smart home, environmental monitoring, and human healthcare monitoring, where efficient energy consumption and security are important issues. Cao et. al. [4] and Zhou et. al. [5] show the recent works for IoT proliferation. Cao et. al. [4] suggests mobility-aware network lifetime maximization for battery-powered IoT applications that perform approximate real-time computation under the quality-of-service (QoS) constraint. Zhou et. al. [5] also provides scheduling tasks onto a heterogeneous multiprocessor system on a chip (MPSoC) deployed to IoT devices for optimizing quality of security under energy, real-time, and task precedence constraints. The blockchain adoption on the IoT devices has a great advantage to a holistic security framework and the efficient device operations. Wang et. al [6] provides a blockchain-based industrial IoT architecture to support immutable and verifiable services. Its proposed architecture seamlessly binds local IoT networks, the blockchain overlay network, and the cloud infrastructure together. Faika et. al. [7] explores blockchain technology for ensuring the communication and data security of IoT devices from malicious cyber-attacks.

IOTA [8], ITC (IOT chain) [9], IoTex [10], and XDAG (dagger) [11] are known blockchain network technologies for IoT applications. IOTA uses a specialized directed acyclic graph (DAG) structure. In an IOTA DAG named Tangle, each transaction consists of one block (referred to as a tip), which is associated with two previous blocks. The associated blocks verify the new block and add it to the IOTA network. To faithfully reflect the characteristics of many IoT applications, the process can be accelerated by a simplified authentication over limited network nodes (see **Fig. 3**).

The Tangle network can be extended in many directions. In the Tangle network, it confirms the block finality through overlapping references. The block finality is calculated by the number of associated blocks. A deeply located block with many associated ancestor blocks receives a higher block-finality score. However, the current IOTA implementation does not faithfully apply this reference overlap. IOTA introduced the concepts of a coordinator and milestones. The coordinator periodically generates milestones. When an added block references a milestone directly or indirectly, it can be assumed to obtain enough references to confirm the block finality. The milestone is a trust device that replaces reference overlapping in the Tangle network [12].

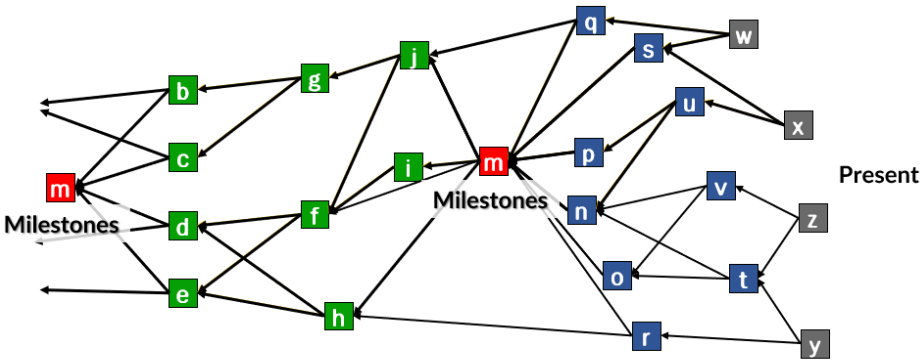


Fig. 3. Tangle Network [8]

The minimum computing requirement of IOTA is more than the average capability of small board H/W for IoT devices. IOTA needs more than 8GB RAM and quad-core CPU power as the minimum requirements while RasberriPi3, the popular small board of IoT devices, provides only 2GB RAM and single-core CPU [13]. In addition, the IRI node in IOTA that provides transaction validation and ledger management needs over the 4GB RAM [14]. Practically, IOTA open source does not provide a sufficient configuration guide to install over the various small board H/Ws. The limitations of IOTA for both of excessive requirements and insufficient applicability make a serious challenge to lightweight blockchain. IoTex limits the data propagation by network scaling. Network scaling is a technique similar to shading, side chains, and inter blockchains. IoTex restricts the data propagation by dividing the network into a single root chain and many sub chains. Only inter-chain transactions run through the root chain. Other local transactions are only distributed inside of the sub chain. The ITC (IoT Chain) also divides the network into a main chain and sub networks. Simple IoT nodes are connected to the head of each sub network. The data can be transferred to the main chain via these heads. XDAG uses the same DAG structure as IOTA; however, it suggests a mining process for network nodes. Note that, the hyperledger fabric is very popular blockchain system for private applications. However, fabric needs a relatively complex structure for lightweight blockchain. The separated nodes such as orderer, peer, endorser, and MSP need a stable network. Much of IoT applications cannot have the stable environments for proper node operations.

The Trusted IoT Alliance was initiated in Berkeley, California. Its objective is to build a trusted IoT ecosystem. The alliance developed an open-source blockchain protocol for over five hundred global blockchain developers and tech companies. Qtum [15], IOTA, IoTeX, IoT Chain, Cyber Physical Chain [16], and Consensys [17] are members of the alliance. The following open projects have been announced to build the ecosystem: smart e-mobility, smart construction, smart building, and smart logistics. The alliance plans to complete a blockchain IoT ecosystem and technologic evolution by the year 2025. As the recent research work, Wazid et. al.[18] proposed a blockchain-based, secure communication scheme for the Internet of Intelligent Things (IoIT). The use of random nonce and timestamps in all exchanged messages protects them against replay and Man-in-the-Middle attacks. The secret keys are not installed directly in memory of any smart device to guarantee the protection from various types of attacks. Also, wazid et. al. show the utilization of blockchain methods (i.e., signature generation and verification procedures) to provide data integrity and authenticity.

The proposed IoT blockchain projects still have many limitations. In IOTA, network collisions occur in the block-verification process. The network scaling of a DAG structure can lead to an excessive burden that reduces the processing speed. The milestones and coordinator are additional unreliable factors. They restrict the distributed nature of the blockchain.

In addition, many recent blockchains (e.g., ITC) use a consensus algorithm called practical Byzantine fault tolerance (PBFT), which is the most advanced form of the consensus algorithm. However, it requires two validation propagation steps and its scalability is difficult to guarantee. In both academia and industry, developers make efforts to secure IoT blockchain technologies that can guarantee more sophisticated network scaling, simpler agreement algorithms, and more convenient application development. **Table 1** shows the qualitative comparison to the representative IoT blockchains (Private Ethereum, IOTA, IoT Chain, and IoTex). The proposed lightweight E2E blockchain platform requires small computing power and memory. It has the flexibility to adopt new lightweight consensus algorithm, such as PoET and RAFT. In addition, it has high modularity to form flexible software packages. However, the lightweight E2E platform has a limitation to expand large scale network. Compare to IOTA, IoT chain and IoTex, which have chain separation capability (i.e., main – sub chain or root – sub chain), It only supports limited inter chain operation to other private chains.

Table 1. Comparison to the Representative IoT blockchain platforms

	Lightweight E2E	Private Ethereum	IOTA	IoT Chain	IoTex
Required Computing Power & Memory	Very Small	Large	Medium	N/A	N/A
Consensus	PoET, RAFT	PoW	Tangle Consensus*	PBFT	PBFT
Scalability	Medium (Inter chain operation support)	Small (Single Chain)	High (Mesh Network)	High (Main/Sub Chain)	High (Root/Sub Chain)
Modularity	High	Low	Low	Low	Low

*)Tangle consensus contains mesh(milestone) referencing between transactions and PoW within transactions

3. Requirements of Lightweight E2E Blockchain Networks

In addition to the small computing power and memory requirements, the essential requirements of an end-to-end lightweight blockchain for IoT applications are as follows:

a. Dynamic consensus: The consensus algorithm should be changeable to build more flexible development environments. PoS (Proof-of-Stake), DPoS (Delegated Proof-of-Stake), PBFT, RAFT, and other novel consensus algorithms are possible candidates for dynamic consensus.

b. Support for the dApp development community: A dApp is the essence of a blockchain service ecosystem. Current dApp development societies should be supported.
c. Highly modular design: The modular system should be able to adopt various transaction rules, permissions, and consensus algorithms.
d. On-chain governance: The configuration and operation rules of the blockchain can be modified by the blockchain network protocols themselves.

Time-based block generation, e.g., PoET (Proof of Elapsed Time)[19] can be a promising alternative to dynamic consensus algorithms. PoET revolves around a distributed leader election across the broadest possible participating population. The cost of controlling this process should be proportional to the value gained during the time elapsed for each device (see **Fig. 4**).

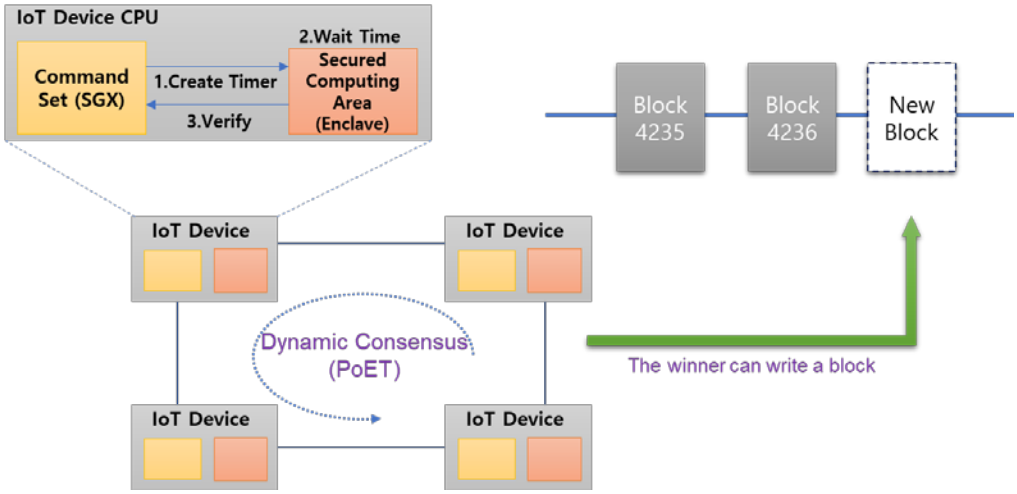


Fig. 4. Dynamic Consensus Algorithm

The dApp developer community provides extensive application diversity. The most popular dApp development environments are Ethereum-based dApp development kits. To guarantee the dApps' compatibility, it needs a middleware concatenating blockchain core for the common user environment, e.g., the web. Seth (Sawtooth-Ethereum) [20] for the Hyperledger Sawtooth platform [21] is a good example of providing Ethereum Solidity compatibility. The Seth middleware provides software compatibility for dApp developers. In addition, software libraries, e.g., web3.js or node.js, should be supported to provide the dApp developers with a convenient development environment (see **Fig. 5**).

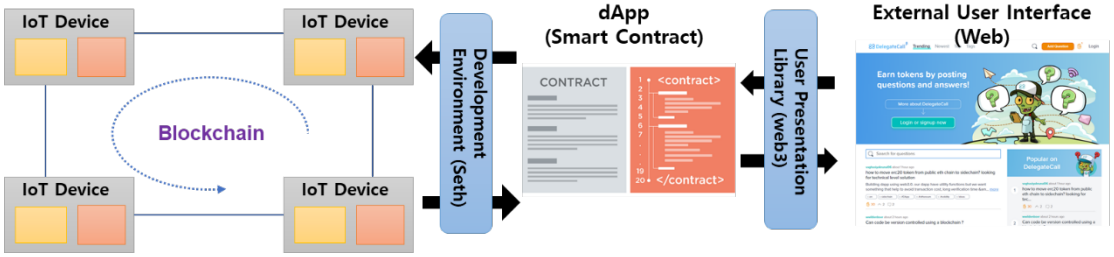


Fig. 5. Support for dApp Development Compatibility

The open-source Hyperledger Sawtooth platform has the advantage of a highly modular design, which makes it easy to modify core mechanisms, e.g., a consensus algorithm. **Fig. 6** shows the Sawtooth implementation structure. The transaction processors are easily added and replaced. Other management modules can be replaced by any customized software modules.

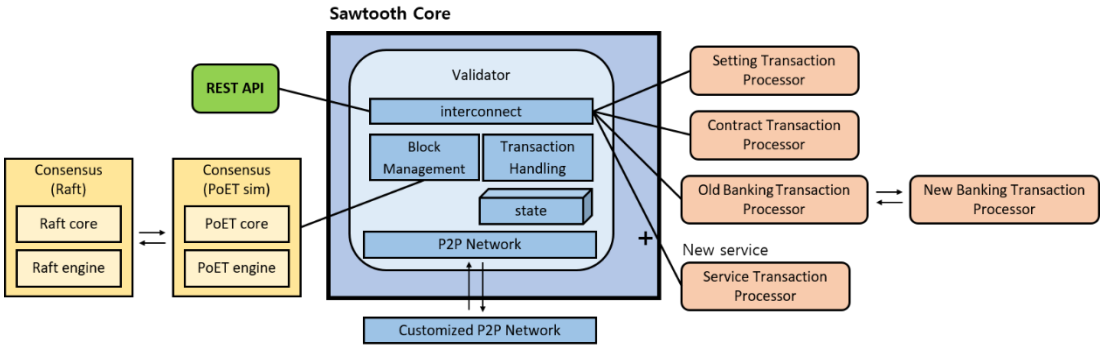


Fig. 6. Modular Design: A Hyperledger Sawtooth Network

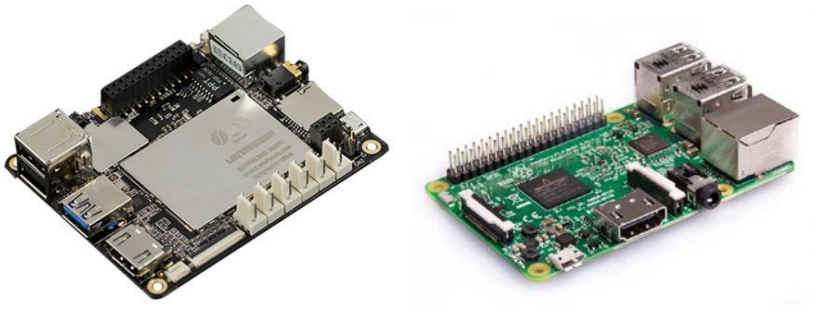
The last requirement is on-chain governance of the blockchain network. On-chain governance is a system for managing and implementing changes in blockchains. In this type of governance, the rules for instituting changes are encoded into the blockchain protocol. Developers propose changes through code updates and each node determines whether to accept or reject the proposed change. If the change is accepted, it is included in the blockchain and baselined. The implementation of the on-chain governance differs between various blockchains. For example, Tezos [22], a cryptocurrency, uses a form of self-amending ledger. DFinity [23], a startup that is using blockchains to build the world's largest virtual computer, unveiled a plan to adopt a hardcoded constitution in its network.

4. Practical Development

4.1 A Lightweight Blockchain for Small Devices

The proposed work built a lightweight blockchain middleware module using the Hyperledger Sawtooth open-source platform. The software structure are reorganized and modified the software modules, e.g., consensus engine, validator, database, and data-serialization functions, for small-scale IoT development devices. Both Intel-architecture IoT devices (e.g., LattePanda)

and ARM-architecture devices (e.g., Raspberry Pi 3) were used for testing. **Fig. 7** shows the two target IoT development devices.



(a) LattePanda (Intel) (b) Raspberry Pi (ARM)

Fig. 7. Tested IoT Development Devices

The Sawtooth open-source platform consists of a validator that creates and maintains a block, a consensus engine, and transaction processors that perform specific functions. This modular structure enables a flexible structure for IoT environments. The validator has a component called a *journal*. The journal uses the consensus engine to create and confirm new candidate blocks. Candidate blocks that satisfy the consensus state are made into a block frame containing only a valid batch ID. (At this time, the candidate block does not include a transaction batch.) The block frame with the valid batch ID is broadcasted to the network through the finalization process. The broadcasted block frame is delivered to the journal's completer via the gossip network for final completion. The completed block is validated in the chain controller and added to the blockchain [21]. **Fig. 8** summarizes the Sawtooth software modules and block processing.

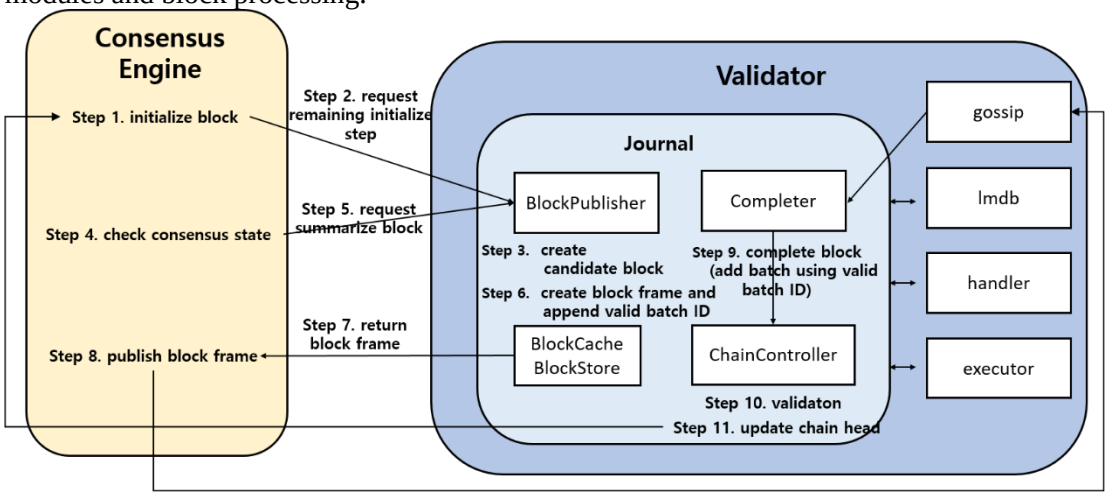


Fig. 8. Block Processing

An essential part of the lightweight blockchain development for IoT devices is support for the ARM architecture. To apply the Sawtooth open-source platform to the ARM architecture, the software libraries are reconfigured to enable them to operate on the ARM architecture. The software modules are also modified, if necessary, to ensure compatibility. For example, the

proposed work has a series of modified LMDB (lightning memory-mapped database) in initializing the genesis process. The execution path resets to guarantee the proper operation of the transaction processors. The data serialization is reconfigured for installing a transaction processor. Finally, we repackage the Sawtooth open-source platform applicable to IoT devices based on both ARM and Intel architectures.

The deployment and maintenance of a blockchain software environment with various devices are key success factors for blockchain dApp distribution. Developers can provide great convenience by applying *Docker* [24] to blockchain deployment. Docker is a container-based open-source virtualization platform. A Docker container is created from a Docker image, which contains the application to be run and its execution environment. Docker guarantees the same execution in various computer environments and can use services through images, without complicated deployment steps [25].

A portable Docker image is created for easy distribution and maintenance. To meet the requirements of various IoT devices, an image of Sawtooth's basic functions should be produced for both Intel and ARM architectures. Docker images make it easy to deploy and operate a lightweight blockchain on various IoT devices (see **Fig. 9**). All software libraries are included in the docker image. We guarantee the software compatibility even in the different operating systems.

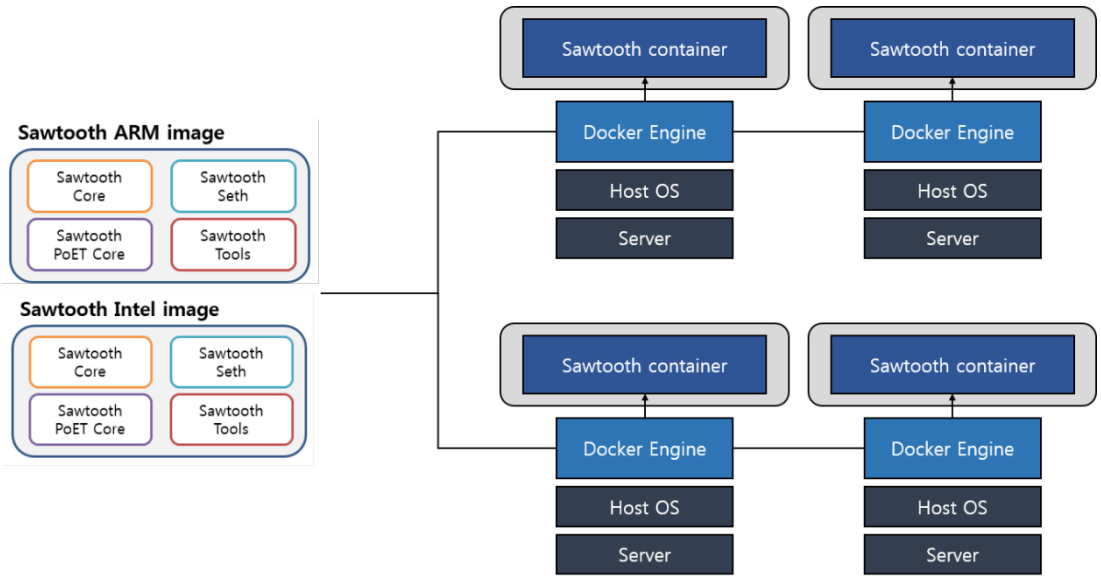
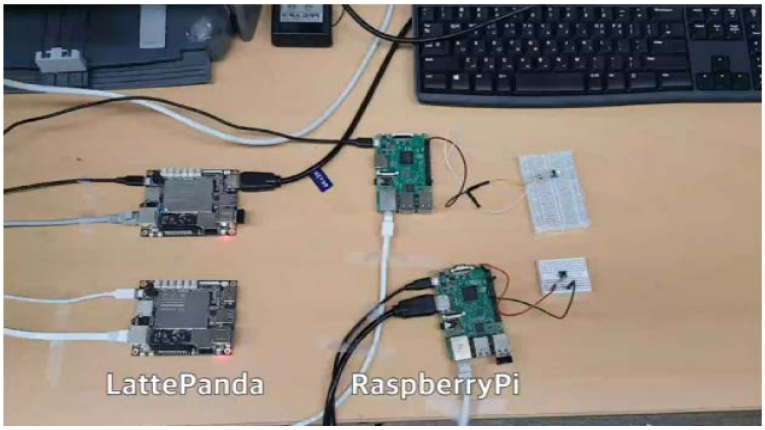


Fig. 9. Sawtooth Docker Image Build

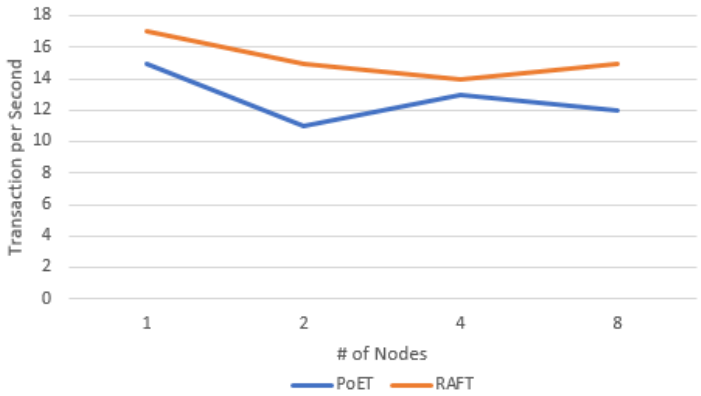
Fig. 10(a) shows the practical implementation of a lightweight blockchain for IoT devices. All transactions are generated from the end devices (Raspberry Pi and LattePanda connected sensors/actuators) and the transactions are recorded in blocks. The block generation is performed in the IoT devices themselves. The end-to-end blockchain for IoT devices is operable as a practical implementation. Note that, the Raspberry Pi uses ARM architecture and the LattePanda uses Intel architecture. Both of two architectures are well applied to our lightweight blockchain.

Fig. 10(b) shows the performance of a currently deployed lightweight end-to-end blockchain network (both of PoET and RAFT). The test environment does not perform very well because

the transaction processors are not optimized. It expects higher performance by suggesting an optimal transaction-processor design, as they can be flexibly configured and modified in the modular design. Note that. our transaction processor is faithfully developed under the guideline of the SDK. The performance can be enhanced by the operation options defined in the SDK. The first option is the batch operation. A transaction batch contains many individual transactions and the transaction validation can be processed batch by batch. The second option is the serial operation. It concatenates multiple transactions to a single transaction. The serial operation also minimizes the number of transaction validations.



(a) Testing Devices



(b) Performance (TPS)

Fig. 10. Tested Example

To show the performance superiority of the proposed E2E IoT blockchain, the results provide the comparison experiments to Ethereum. Other IoT blockchain solutions, such as IOTA or IoT chain do not provide a lightweight blockchain software module: the IOTA needs at least 8GB RAM and quad core CPU for a practical adaptation. Even worse, the source codes of IOTA or IoT chain are incomplete to apply the general end devices, such as RasberriPi (ARM) or LattePanda (Intel). Ethereum is the almost only one to show the performance difference of the developed lightweight E2E blockchain solution. However, Ethereum has a limitation for direct comparison to the developed solution. Because the computation difficulty control

depends on the network size, it must assume the experimental network size and decide the computational difficulty of Ethereum. Note that, the developed solution has a fixed computing resource requirement regardless of network size. Thus, it assumes the experiment network size to 10 for comparison. The **Fig. 11(a)** shows a general performance metric, TPS (Transaction per Second). Ethereum has slightly higher performance. However, our developed solution has the advantages on latency (**Fig. 11(b)**), CPU consumption (**Fig. 11(c)**) and memory consumption (**Fig. 11(d)**). Especially, for memory consumption, our solution only has 1/3 requirement compared to Ethereum. Even in TPS, the performance of Ethereum cannot provide same level when it is applied to greater network size. It expects superior performance in TPS when we apply our solution to a greater network.

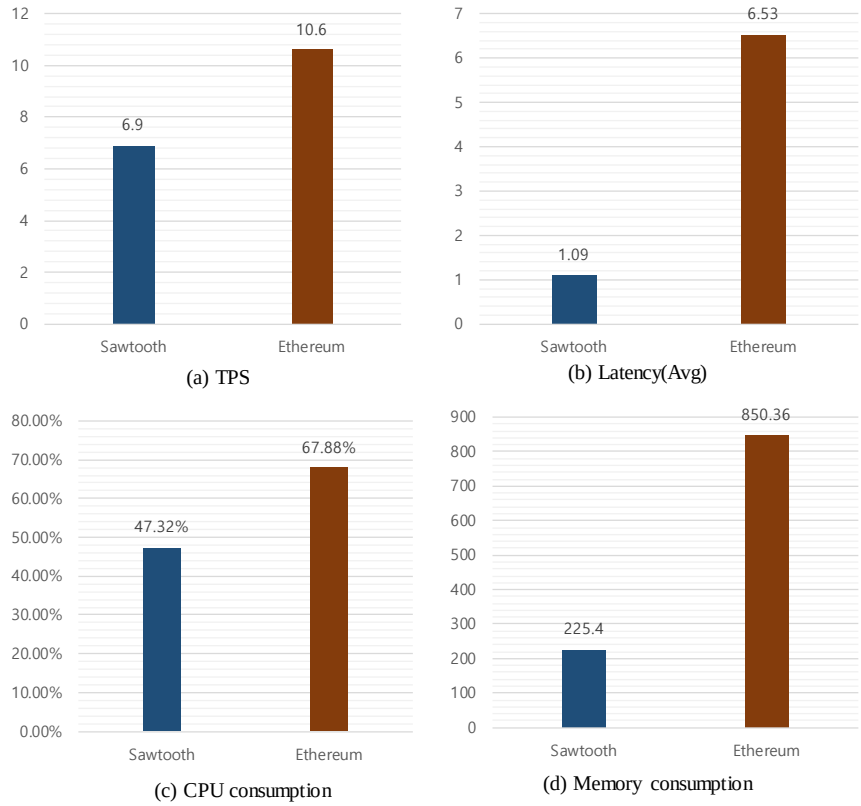


Fig. 11. Comparison between Ethereum and Sawtooth

4.2 IoT Applications

We have developed a food-growing–environment monitoring and certification dApp to prove the usability of the proposed lightweight end-to-end blockchain network. The current certification system grants the certification mark after inspection by governmental agencies. However, pesticides or hazardous substances are occasionally detected, even in certified foods. The proposed work reconstructed the current unclear certification system into a trusted process using a blockchain dApp [26][27]. pH sensors and HCHO (formaldehyde) sensors were used to monitor the food-growing environment. **Fig. 12** shows the developed monitoring/certification model.

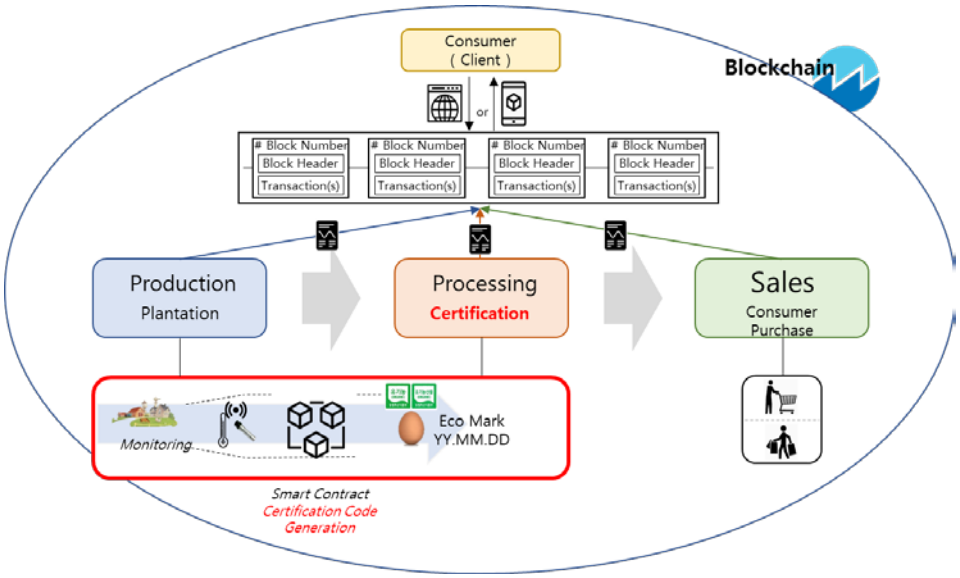


Fig. 12. Food Certification Model

Fig. 13 shows an example of a farm. Each farm has an identifier (e.g., farmId). In the example, three areas are built with identical sensor sets. A single area generates a single product identifier (e.g., productId). The productId of an area changes with every product-certification cycle. If the certification is denied because hazardous substances are detected, it scraps the products and start a new certification cycle.

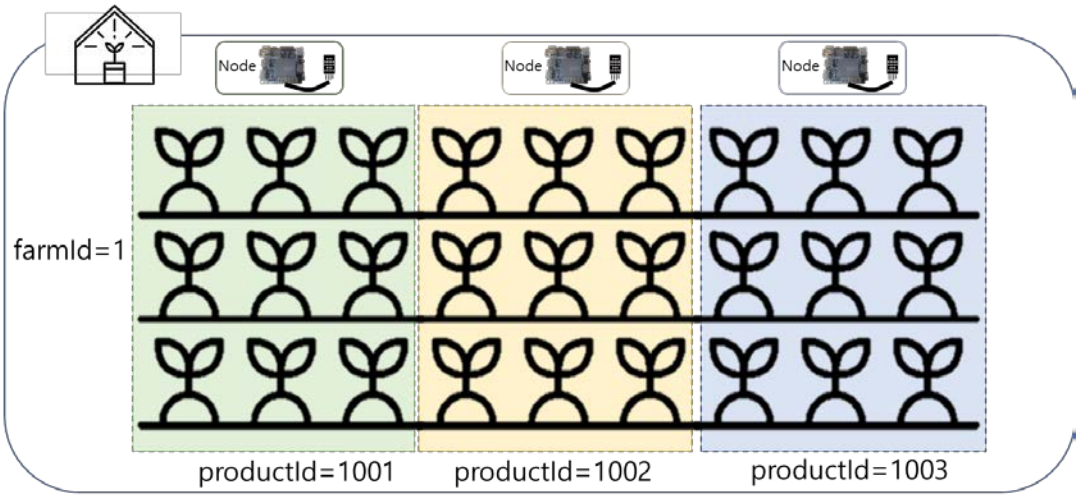


Fig. 13. Example Farm

Hyperledger Sawtooth implements the dApp by defining a *Transaction Family*. The transaction family contains the contents necessary to build a *Transaction Processor* that has the dApp's action codes. The transaction processor defines the logic part of the dApp. The transaction processor connects to a validator for transaction-payload processing and state-variable changes. The developed dAPP also defines a data model and use it as a structure to record data.

Clients build and sign the transactions. The client also sends a batch of transactions to the validators. The client posts the batches via the REST-API, or connects directly to the validator using an asynchronous messaging library (ZeroMQ). The Sawtooth dApp supports Python, JavaScript, and Go language-based SDKs for building dApps. The *Ecosupply* dApp has the custom designed transaction family using Python. **Fig. 14** shows the designed transaction-family components.

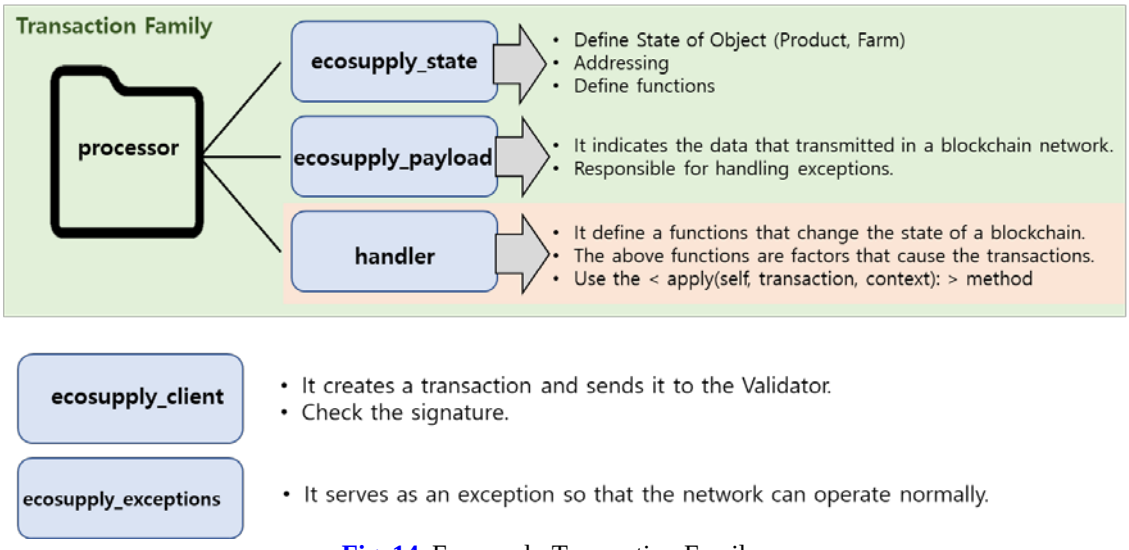


Fig. 14. Ecosupply Transaction Family

Ecosupply_state presents the state information of the dApp objects. The Ecosupply dApp updates the states for two dApp objects: product and farm. **Table 2** shows the attributes included in the state of each object.

Table 2. States of Objects (ecosupply_state)

Object	Attribute	Description
Product	productId	Product identifier
	areaId	Area identifier. Each area has a single sensor set
	farmId	Identifier of the farm to which the product belongs
	sensor1	pH sensor measurement
	sensor2	HCHO sensor measurement
	certification	Certification confirmed = True Certification denied = False
	date	Date of product harvesting
Farm	farmId	Farm identifier
	address	Physical address of the farm
	penalty	Number of denied certifications

The two objects have blockchain account addresses. The blockchain account address is generated from a SHA512 hash function. The account addresses of the product and farm are assigned as follows:

- Product account address = sha512('ecosupply')[0:6] + sha512(productId)[0:64]
- Farm account address = sha512('ecosupply')[0:6] + sha512(farmId)[0:64].

Note that the productId and farmId should be different to separate the product account address and the farm account address.

The transaction message format is defined in ecosupply_payload. The transaction message includes the actual data transfer in the blockchain networks. The handler transaction family activates the logic part of the dApp. The application functions, e.g., create_farm, create, delete, certificate, show, and list, are defined in the handler. **Fig. 15** shows the connection between the ecosupply transaction processors and a validator. The transaction processor uses ecosupply transaction families to implement the designed dApp. When the connection completes, Sawtooth blockchain-network nodes can activate the transaction processors.

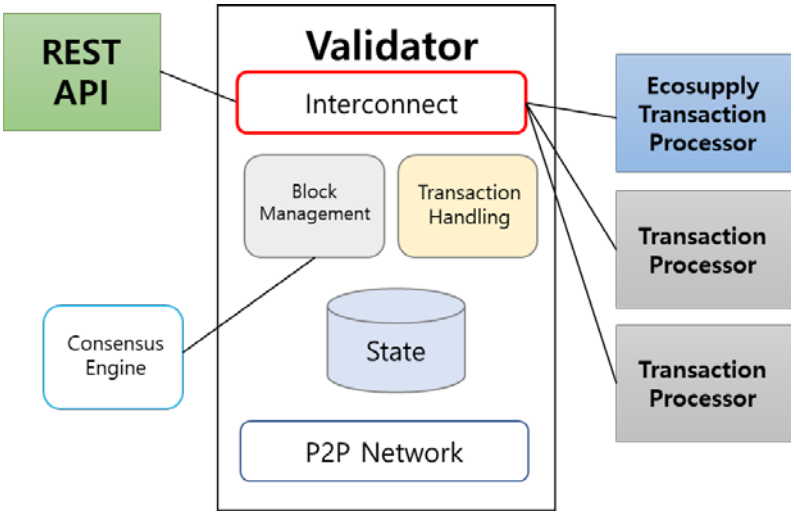
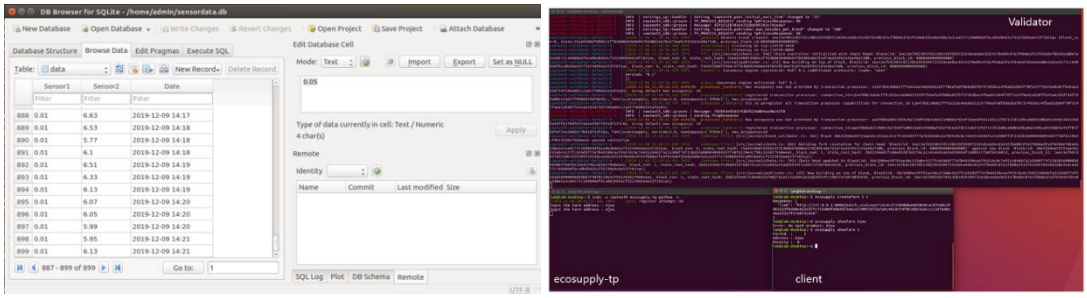


Fig. 15. Connecting the transaction processor to the validator

The testbed software uses the Ubuntu 16.04 LTS operating system, Sawtooth version 1.1.5, and Docker 19.03.1 / Docker-compose 1.24.1. The blockchain consensus mechanism uses PoET-Simulator. The LattePanda boards use 2 GHz processors and 32 GB RAM. The LattePanda has a built-in Arduino Leonardo microcontroller, which writes and compiles sensor-related code using the Arduino IDE. The sensor values needed for the certification are stored in an SQLite database in real-time. SQLite was selected for its simple installation and deployment, and it is installed automatically with the Python language. In addition, its small size and simplicity make it suitable for storing sensor data.

When a certification transaction occurs, the sensor values are read from the database and the current states of the products are updated. A runtime dApp creates a farm with a farmId and a product object with a productId. When a certification transaction is generated for each product, the state is changed by taking sensor data from a database. Ten certification cycles are performed. When all 10 cycles return with the certification confirmed (i.e., certification is true), it can release the certified products. If even a single cycle is determined to be false, the product is discarded.

Fig. 16 shows the dApp operations in terminal mode. Sensor data are stored in the distributed ledger (**Fig. 15(a)**). The developed transaction processor is activated in the validator (**Fig. 15(b)**). The terminal window for transaction processor presents the farmId, productId, sensor values, certification information, and time stamps. Users can check the certification of each product using the displayed information. To provide a user-friendly interface, the terminal mode dApp is transformed into a web framework (**Fig. 15(c)**).



(a) Sensor data storing on Ledger (b) Transaction Processor Activation on Validator

(c) Certification on Web Page

Fig. 16. Operation Results

The **Fig. 17** shows the details of whole blockchain and dApp operations in food-growing environment. The first step is sensing data collection. The sensors installed in farm periodically report monitoring data such as hazardous chemicals. The second step is block builder selection. The monitoring data should be recorded in the distributed ledger by the selected block builder. The proposed work can apply PoET or RAFT as a consensus on the block builder selection. The selected block builder writes the transaction on the ledger in the third step. The block builder reads the state information for monitoring data and updates the state with time stamps. Then, the block builder makes a transaction tree and generates a Hash value for the transaction tree. The hash value is recorded in a block. The block builder transfers the block to the other validators. The fourth step is operated by the dApp. The dApp checks the validity of sensor monitoring data and decides the certification whenever it receives user requests for certification.

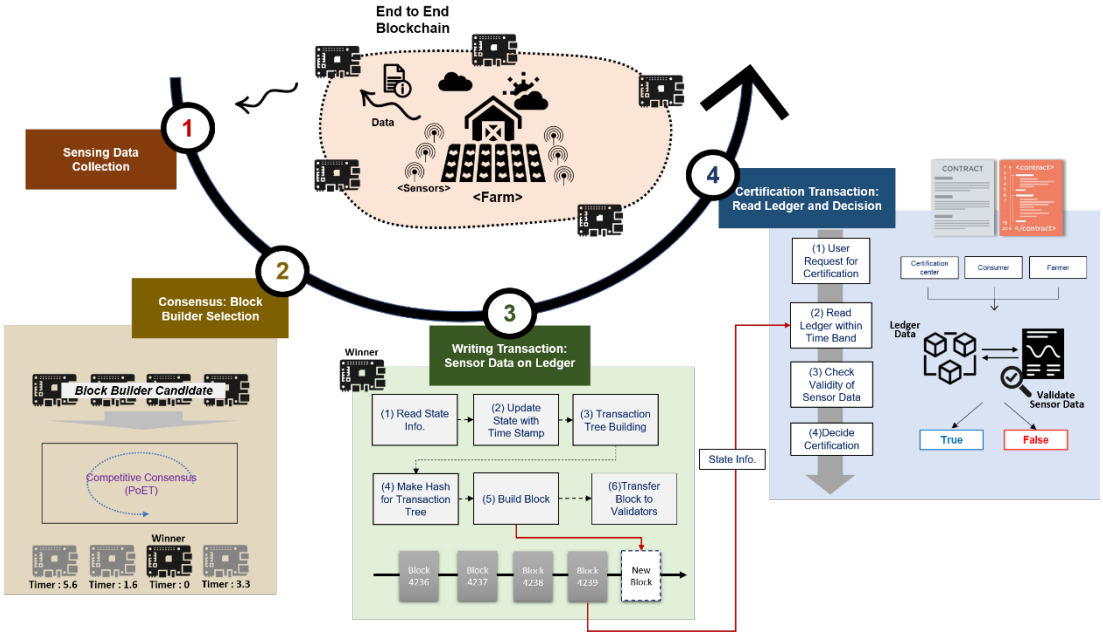


Fig. 17. Development Details

5. Conclusion

We designed a lightweight end-to-end blockchain network that applies to common IoT applications. Its enhanced modular architecture and lightweight consensus mechanism guarantee its practical applicability for general IoT applications. The modular architecture is easily customized to address various application fields. The developed food-growing environment uses the modular architecture to build farm installed devices and user application to make a certification to products. In addition, the proposed blockchain network is highly software compatible because it adopts the Hyperledger development environment. Directly embedding the proposed blockchain middleware platform in small computing devices proves its practicability. The highly compatible blockchain middleware platform ensures the end-to end secure transaction transfer. It eliminates the oracle risk of typical blockchain platform. An end-to-end IoT blockchain exponentially expands the distributed-data reliability. From the sensors, the sensing data were transmitted in the form of blockchain transactions and the transactions were recorded in blocks. The actuation was performed following the results of the dApp operation. For a certain condition, the dApp generated action codes for the actuators. Third-party applications can use the data collected/stored by other dApps for developing their applications. The current IoT blockchain does not yet provide a flawless service platform. However, the end-to-end IoT blockchain will be a game changer in future industry. Continuous trials and developments will complete a ready-to-market IoT blockchain. Our lightweight blockchain module is easily applied to various applications. Especially, the massive IoT applications, one of the essential 5G user applications, provide the best applicability of our lightweight blockchain. Various small IoT devices embedded lightweight blockchain module initiate a smart contract for bandwidth sharing in 5G infrastructure, such as mobile edge clouds, to negotiate between stakeholders.

References

[1] V. Buterin, “A next-generation smart contract and decentralized application platform,” *White Paper*, pp.1-36, 2014. Article (CrossRef Link).

[2] “EOS.IO, Technical White Paper,” *Github*, 2018. Article (CrossRef Link).

[3] Linux Foundation, Hyperledger Project. Article (CrossRef Link).

[4] K. Cao, G. Xu, J. Zhou, M. Chen, “QoS-Adaptive Approximate Real-Time Computation for Mobility-Aware IoT Lifetime Optimization,” *IEEE trans. on Computer-aided Design of Integrated Circuit and Systems*, Vol.38, No.10, pp. 1799 – 1810, 2019. Article (CrossRef Link).

[5] J. Zhou, Z. Liu, X. Zhou, “Security-Critical Energy-Aware Task Scheduling for Heterogeneous Real-Time MPSoCs in IoT,” *IEEE Trans. on Service Computing*, *Early Access*, vol.13, no.4, pp..745-758, 2020. Article (CrossRef Link).

[6] G. Wang, Z.J. Shi, M. Nixon, S. Han, “ChainSplitter: Towards Blockchain-based Industrial IoT Architecture for Supporting Hierarchical Storage,” in *Proc. of IEEE International Conference on Blockchain*, 2019. Article (CrossRef Link).

[7] T. Faika, T. Kim, J. Ochoa, M. Khan, S. Park, C. S. Leung, “A Blockchain-Based Internet of Things (IoT) Network for Security-Enhanced Wireless Battery Management Systems,” in *Proc. of IEEE Industry Applications Society Annual Meeting, Baltimore, USA*, 2019. Article (CrossRef Link).

[8] S. Popov, "The tangle," *cit. on*, pp.1-28, 2018. Article (CrossRef Link).

[9] IoT Chain: A high-security lite IoT OS White Paper. Article (CrossRef Link).

[10] IoTeX-A Decentralized Network for Internet of Things. Article (CrossRef Link).

[11] “XDAG: A new DAG-based cryptocurrency White Paper,” *Github*, 2018. Article (CrossRef Link).

[12] S. Popov, “The Coordicide,” 2020. Article (CrossRef Link).

[13] IOTA, “Setup a Private Tangle,” Article (CrossRef Link).

[14] IOTA, “Run IRI,” Article (CrossRef Link).

[15] Alex Norta, “Smart-Contract Value-Transfer Protocols on a Distributed Mobile Application Platform,” 2017.

[16] “CPChain White Paper 2.0,” 2019. Article (CrossRef Link).

[17] “Consensys White Paper,” 2017. Article (CrossRef Link).

[18] M. Wazid, A. K. Das, S. Shetty and M. Jo, "A Tutorial and Future Research for Building a Blockchain-Based Secure Communication Scheme for Internet of Intelligent Things," *IEEE Access, Early Access Online Version*, vol. 8, pp. 88700-88716, 2020. Article (CrossRef Link).

[19] PoET 1.0 Specification. Article (CrossRef Link).

[20] Linux Foundation, Hyperledger Sawtooth Seth. Article (CrossRef Link).

[21] Linux Foundation, Hyperledger Sawtooth Project. Article (CrossRef Link).

[22] L.M Goodman, "Tezos — a self-amending crypto-ledger White paper," pp.1-17, 2014. Article (CrossRef Link).

[23] T. Hanke, M. Movahedi, and D. Williams, "Dfinity technology overview series, consensus system,” *arXiv preprint arXiv:1805.04548*, 2018. Article (CrossRef Link).

[24] C. Boettiger, "An introduction to Docker for reproducible research,” *ACM SIGOPS Operating Systems Review*, 49(1), pp71-79, 2015. Article (CrossRef Link).

[25] D. Bernstein, “Containers and cloud: From lxc to docker to kubernetes,” *IEEE Cloud Computing*, vol. 1, no. 3, pp.81-84, 2014. Article (CrossRef Link).

[26] F. Tian, “A Supply Chain Traceability System for Food Safety Based on HACCP, Blockchain & Internet of Things,” in *Proc. of International Conference on Service System and Service Management, Dalian, China*, 2017. Article (CrossRef Link).

[27] M. Nakasumi, “Information Sharing for Supply Chain Management based on Block Chain Technology,” in *Proc. of IEEE 19th Conference on Business Informatics, Thessaloniki, Greece*, 2017. Article (CrossRef Link).

Seungcheol Lee received the B.S. degree in industrial engineering from Ajou University. Currently, he is a M.S. student in Ajou University, majoring Industrial Engineering. His interest includes Blockchain platform and applications of machine learning. He is a researcher of Artificial Intelligent IoT Lab in Ajou University.

Jaehyeon Lee received the B.S. degree in Consumer Science from Chungnam National University. He received M.S. degree in data science from Ajou University. He has developed a Blockchain service. His interest includes Blockchain dApp and applications of machine learning.

Sengphil Hong received the Ph.D. degree in Computer Science from the Korea Advanced Institute of Science and Technology (KAIST) in 2003. He also received M.S. degree in Computer Science in Ball State University, Indiana, U.S.A. He received B.S. degree in Indiana State University, Indiana, U.S.A. His interests include security in computer systems and blockchain platforms. He has developed various computer security frameworks. Also, he has experience on the state policy of computer systems. He served as a senior consultant in LG CNS, Korea and a faculty member of Sungshin Womens University, Korea. Currently, he is a CEO of Hancom WITH, Korea.

Jae-Hoon Kim received the B.S., M.S., and Ph.D. degrees in Management Science from the Korea Advanced Institute of Science and Technology (KAIST) in 1996, 1998, 2003. His interests include IoT networks, blockchain platforms ubiquitous networks. He has developed various network operation frameworks and simulation test-beds. Also, he has experience on the mobile service design and strategy. He served as a system architect of wireless systems in SAMSUNG Electronics and a system engineer in SK Telecom, Korea. Currently, he is now a faculty member of Industrial & Information Systems Engineering in Ajou University, Korea. He is a member of the IEEE Communications Society and the IEEE Vehicular Technology Society.