Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 5758741, 15 pages
https://doi.org/10.1155/2018/5758741

Research Article

A Peer-to-Peer Architecture for Distributed Data Monetization in Fog Computing Scenarios

Francisco de la Vega,[1] **Javier Soriano**,[1] **Miguel Jimenez**,[1] **and David Lizcano**[2]

[1]*School of Computer Science, Universidad Politécnica de Madrid, 28660 Madrid, Spain*
[2]*Madrid Open University, 28400 Madrid, Spain*

Correspondence should be addressed to Francisco de la Vega; fdelavega@fi.upm.es

Received 15 June 2018; Accepted 22 July 2018; Published 4 September 2018

Academic Editor: Raquel Lacuesta

Modern IoT deployments do require considerable investments that might only be justified if the data being gathered could be monetized, which leads to the need for a digital data marketplace. In many cases, the provider of the IoT data needs to process it locally for data curation, aggregation, stream processing, etc. At the same time, the consumer could be interested in nearby data. This scenario resembles a fog computing architecture where companies require being able, keeping data under their control, to securely make it available to other companies in a peer-to-peer fashion, without needing a cloud intermediary (like traditional marketplaces do), thus maximizing the locality of the processing and avoiding the existence of a bottleneck when the intermediary makes the data delivery for accounting purposes. Nevertheless, this imposes a hard requirement: by not having a central marketplace, the peers (seller and customer) need to trust each other, which, in turn, requires enforcing a nonrepudiation schema. In this paper, the authors propose a distributed peer-to-peer architecture for such a data marketplace that takes advantage of the architectural fundamentals of fog computing, in which data processing, filtering, and stream based event generation is done in a fog node along with the data, and where relationships, both commercial agreements and data delivery, are performed directly between producers and consumers without the need of mutual trust thanks to the usage of blockchain principles (e.g., distributed ledger, consensus mechanism). The proposed architecture is validated through a case study involving a set of key issues regarding nonrepudiation commonly identified when moving from a centralized marketplace to a distributed one. Moreover, it is shown that the proposed solution does not bring in any limitation with regard to a centralized marketplace solution, in terms of pricing models (subscriptions, pay-per-use, etc.) or usage conditions (contract duration, updates rate, etc.).

1. Introduction

Data is becoming one of the most valuable assets, being considered even more important than oil ([1, 2]). This statement applies to many data sources, including those generated with IoT sensors. Many companies which have modern IoT deployments do require considerable investments that might not be justified for the expected revenue due to the exploitation of the generated information internally by the owner. This may generate reluctances among companies willing to exploit that information, especially when only a portion of that information is directly profitable for the company. Besides, such IoT deployments generate huge amounts of data that might be interesting both for companies in the same sector (competitors) and for companies on other sectors but that do benefit from such information. In the latter case, performing the deployment by themselves is even more unlikely. In many cases, the provider of the IoT data needs to process it locally for data curation, data aggregation, and event generation based on stream processing, etc. At the same time, the consumer could be interested in nearby data, leading to a scenario that resembles fog computing ones.

An example of such scenario is a company owning the air quality sensors deployed along a city in multiple fog nodes, each of them making local processing for the above-mentioned purposes, and a number of smart buildings distributed all along the city, requiring the data from the closest fog node to feed the algorithms controlling their

smart systems (e.g., free cooling operation, air conditioning optimization, predictive maintenance of air filters, and user safety protocols).

This scenario introduces the need of a digital marketplace to satisfy both interests: the owner of an IoT deployment being able to monetize data by selling it, and other companies being able to leverage on that data to make their businesses or accomplish their goals.

These interactions have traditionally taken place on electronic marketplaces [3] which serve as central markets to integrate offerings from multiple sellers providing not just a product catalog for search, discovery, and comparison [4], but also transaction support in terms of negotiation, contracting, and settlement [5]. Traditional marketplaces, besides, represent a central point of failure, an interaction bottleneck and do play a special role in between sellers and consumers, who both need to trust him.

To overcome this limitation, [6] proposes distributing the marketplace as a peer-to-peer network. In such a marketplace, resembling a fog computing architecture as requested in [7], there is no need of a cloud intermediary (like traditional marketplaces do [8]), thus maximizing the locality of the processing and avoiding the existence of a bottleneck when the intermediary makes the data delivery for accounting purposes (e.g., Apigee). In addition, companies are able on their own (under their control) to securely make those data available to other companies when using a peer-to-peer fashion. Nevertheless, this imposes a hard requirement: by not having a central marketplace, the peers (producer and consumer) need to trust each other, which, in turn, requires enforcing a nonrepudiation schema.

Authors in [6, 9] propose the usage of blockchain principles [10] (e.g., distributed ledger, consensus protocol, and public key cryptographic system) to manage trust on peer-to-peer distributed marketplaces.

In this paper, the authors propose a distributed peer-to-peer architecture which takes advantage of the architectural fundamentals of fog computing, in which data processing, filtering, and stream based event generation is done in a fog node along with the data, and where relationships, commercial agreements, data delivery, access control, and access log are performed directly between producers and consumers without the need of mutual trust or central role, thanks to the usage of blockchain principles. The proposed architecture is validated through a study case involving a set of key issues regarding nonrepudiation commonly identified when moving from a centralized marketplace to a distributed one. Moreover, it is shown that the proposed solution does not bring in any limitation with regard to a centralized marketplace solution, in terms of pricing models (subscriptions, pay-per-use, etc.) or usage conditions (contract duration, rate of data updates, etc.).

2. Related Work

As explained, blockchain technology has been proposed for distributing data marketplaces. Besides, some marketplace functions have also been implemented using distributed ledger technologies and are relevant for the design of such a distributed marketplace, such as how to distribute the data or how to control the access to it considering privacy concerns.

For data distribution using blockchain technology, some authors propose using an off-blockchain storage based on distributed hash tables (DHT), where links to reallocation of data are encrypted inside blocks [11]. This scheme is replicated on [12] for healthcare data sharing and on [13] for building a trackable and reputable distributed file system called InterPlanetary File System (IPFS). The benefit is the off-loading of the distributed ledger of the data itself, maintaining access-control and integrity capabilities. This distribution mechanism, however, does not allow several accounting schemes required for usage-based price models, such as volume of information accessed.

Privacy management and access-control of shared data are also tackled using blockchain technology, especially on the medical sector for privately sharing Electronic Medical Records (EMRs) with institutions other than the ones that generate such information, and giving the patient the control of her data. The works [14, 15] are relevant proposals using this scheme. And regarding IoT data sharing, [16] proposes fine-grained smart contracts based access-control scheme.

Regarding the monetization of IoT data, there are purely centralized proposals such as [17], marketplace centralized both on the data catalog and on the exchange itself where IoT data providers register their offers and help the consumers finding them. There is also a commercial centralized cloud-based marketplace call Terbine (http://www.terbine.com/) offering high control on how IoT data can be used. The paper [18] gives one step towards the decentralization by empowering data providers with the ability to define sharing preferences and data privacy and deliver the data to consumers in a peer-to-peer fashion, whereas the data marketplace where their offers are published is centralized.

There are also some approaches that propose a distributed solution for the creation of an IoT data marketplace. Wörner does present on [19] (which extends [20]) a prototype of a decentralized data market directly payable API endpoint for peer-to-peer data exchange and payments, based on Bitcoin micropayments. OpenBazaar proposes peer-to-peer marketplaces to directly connect data providers and consumers using Bitcoin as their digital cryptocurrency. Micropayments, however, have turned into unfeasible with cryptocurrencies whose transaction fees have dramatically escalated such as Bitcoin or Ethereum [21]. Latest proposals are not tied to any specific cryptocurrency or payment method such as IoT layer, a commercial blockchain-based security layer for direct access to IoT devices with minimal access-by-payment options, or Ocean Protocol [22], a recent decentralized blockchain-based marketplace for data distribution with a focus on Artificial Intelligence and services execution on the data location.

3. Case Study and Requirements

The case study presents a distributed data marketplace and deals with the advertisement and acquisition of data between two untrusted peers of the network, each of them having their own fog node for local data processing. *Company A*

provides smart city services on different verticals. As part of these services, Company A owns a set of air quality sensors along different neighborhoods of the city, which generate raw data, grouped together in a set of fog nodes each of them capturing, curating, and processing raw data of a particular geographic area. Within each of this fog nodes, raw sensor data is fine-tuned, aggregated, and processed to generate high level information. Besides, CEP (Complex Event Processing) is also performed at local level in order to detect anomalous situations. Finally, processed data is sent to company facilities for performing big data processing at a city level.

Within this scenario, *Company A* realizes that part of the data produced in the different nodes could be monetized and shared with interested peers, generating an extra profit for the company, under certain terms and conditions, such as not using such data for creating competing solutions, or not reselling it.

Company B owns a smart building which includes a set of sensors and actuators for the optimization of their HVAC systems (heating, ventilation, and air conditioning), controlling locally factors such as temperature, humidity, water flows, pump speeds, and fan speeds in order to automatically maintain proper conditions while optimizing the consumption of the systems.

In the described scenario, *Company B* decides to improve their HVAC management by incorporating a free cooling system, which optimizes the consumption using low external temperature levels to regulate building climatization by incorporating air from the outside into the HVAC system. However, this introduces extra control requirements of pollutant levels to avoid unhealthy conditions and maintain air filters, requiring a stream of air quality information of the particular area. This data stream can be acquired from the node that Company A owns in the area.

3.1. High Level Requirements. The proposed use case has some high level requirements that the system used for the data sharing and monetization has to satisfy in order to be successful. First of all, there must exist interoperability between the fog nodes; that is, one fog node should be able to understand the format and the protocol of the data streams of the other participants to be able to incorporate this information into its processes.

Additionally, a data marketplace able to advertise data streams, managing pricing, and usage terms has to be incorporated. This system has to be able to grant access to acquired data and make accounting of the consumption in order to support pay-per-use and validate that terms and conditions are satisfied. However, for the proposed scenario where the different participants process data locally in a fog node, a centralized approach is not suitable, since it introduces the need for an intermediary playing an special role (which the participants have to trust on) and represent a central point of failure.

Distributing the marketplace introduces the need to trust about the validity of published offerings, the signed agreements between providers and consumers, and the data requested and interchanged among peers. In a centralized marketplace, it acts as 3rd party performing the first two of them, provided that the peers trust the marketplace, but the former is not easily realized.

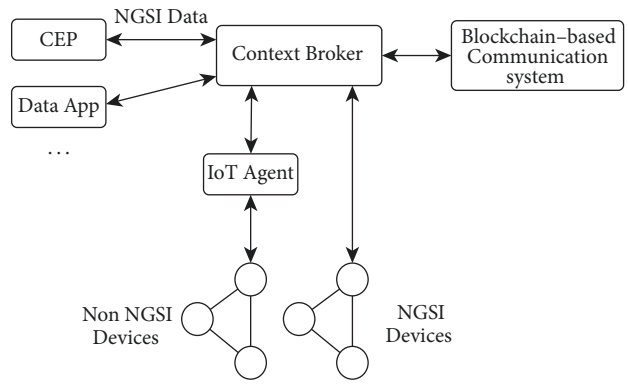

Figure 1: Local node architecture for data processing.

4. Proposed Solution

In this paper, the authors present a distributed peer-to-peer data marketplace in a fog computing scenario, enforcing trust and nonrepudiation among peers. Trust is performed by using distributed ledger, leveraging blockchain technology for immutable and nonrepudiable advertisement of offerings, sign of agreements and for accounting on the data requested and sent. We propose a fog computing architecture where marketplace peers are fog nodes providing or acquiring data, where nodes based on FIWARE (https://www.fiware.org) technology hold their generated and acquired data for local computation, relying on blockchain technologies for the acquisition and interchange of data.

4.1. FIWARE Fog Computing-Enabled Architecture. FIWARE is an open standards-based platform leveraging an open, public, and royalty-free architecture and a set of open specifications that allow developers, service providers, enterprises, and other stakeholders to develop and deploy innovative products and digital services in a number of application domains, including Smart Cities and Industry 4.0.

The proposed solution relies on FIWARE technologies for the interoperability of data within the fog nodes. The FIWARE platform uses the FIWARE NGSI v2 (an enhanced version of Open Mobile Alliance NGSI) and defines a set of Data Models (https://www.fiware.org/developers/data-models/) adding common syntax and semantics for different verticals. Currently, around 90 cities from 19 countries in Europe, Latin America, and Asia-Pacific have signed up the Open and Agile Smart Cities (OASC) alliance (http://oascities.org) membership meaning they intend to share and publish their smart city data by means of FIWARE technology.

Figure 1 depicts the architecture of the FIWARE-enabled fog node proposed by the authors. As can be seen, data is modelled as NGSI entities, according to some data model, and managed by the FIWARE Context Broker. Context Broker enables on-demand context information management based on the FIWARE NGSI open specification, including

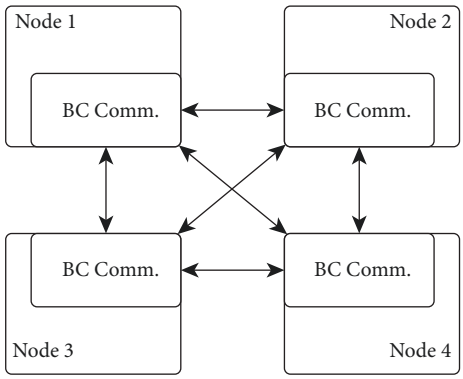

Figure 2: Blockchain-based peer-to-peer network.

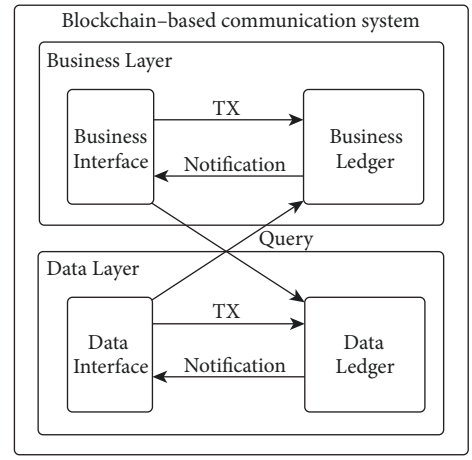

Figure 3: Blockchain-based communication system.

both queries (even geographic-based) and subscriptions for asynchronous notifications on changes. The Context Broker holds the current value of all the different entities managed by the platform being updated with the information provided by the different *IoT agents*, which are isolated from the IoT layer specifics. For data processing, a Complex Event Processing FIWARE component can be connected to the Context Broker. Finally, the Context Broker provides a federation mechanism which enables creating data registrations and forwarding queries and subscriptions, supporting the execution of local apps using Context Broker data, irrespectively of the actual location.

For the exploitation of the data, local *Data Apps* connect to the Context Broker and query/subscribe for the entities using the NGSI v2 API, irrespectively of the source of such entities, local or remote. If data requested by Data Apps comes from data sets acquired in the marketplace, the blockchain-based communication mechanism would request the information obeying agreements, generating access and usage logs, and retrieving the required date, as described below. Besides, for the implementation of the marketplace concepts, we have taken as basis the Business API Ecosystem FIWARE GE, implementing its business concepts.

4.2. Blockchain-Based Communication System. The distribution of the marketplace implies interactions among nodes, which act as sellers, customers, or both, in the publication of offerings, in the establishment of agreements, and in the exchange of the data itself. Such distribution is performed using blockchain technology, as shown in Figure 2, which provides not only the distributed storage, but also the privacy, security, and trust of the distributed marketplace.

Given the nature of commercial transactions, our proposal relies on a permissioned blockchain solution, where participants are identified, linking the marketplace with legal guarantees of the real world. Such permissioned scheme is implemented as a certificate hierarchy, delegating the participation in the network to a main Certification Authority, which must sign the node certificates, therefore delegating on the nodes the issuing of user certificates.

The blockchain-based communication system (Figure 3) deals with the data sharing and monetization capabilities at two different layers: (1) the Business Layer, which manages all the aspects related to data advertising, location, and monetization, and (2) the Data Layer, which deals with the data sharing and accounting capabilities. Each of these layers includes a distributed ledger which stores the associated information in the form of immutable transactions, the business logic dealing with such transactions, and an asynchronous REST API. These APIs hide from the complexity of the blockchain technologies, offering data interchange and business high level actions, managing the corresponding creation and monitoring of transactions.

The split on two different ledgers is due to the different requirements, in terms of throughput, delay, security, and functionality, exposed by each layer. In particular, the business layer requires strong participant identification, transactions to be validated before its data can be accessed, and support for smart contracts for validation and setting up of agreements, not imposing big requirements on throughput and delay. On the other hand, the Data Layer requires transaction information to be available as fast as possible in order to avoid an excessive delay in the consumption of the acquired data, as well as the best possible throughput. Taking into account these requirements, the proposed solution uses two different distributed ledger technologies:

(i) The Business Layer uses Hyperledger Composer (https://www.hyperledger.org/projects/composer) on top of Hyperledger Fabric (https://www.hyperledger.org/projects/fabric) in order to create a permissioned network composed only of the peer and orderer nodes deployed by the participants of the peer-to-peer network proposed by our solution. Hyperledger Composer introduces abstraction level that enable defining types of transactions and its attached smart contracts to manage a set of assets. These assets can be created, modified, or deleted by the smart contracts of the transactions, composing the *world state* (https://hyperledger-fabric.readthedocs.io/en/master/ledger/ledger.html), which is a database with latest state of every asset whose consistency is maintained

by the transactions stored in the ledger. In addition, Hyperledger Composer defines an ACL-based mechanism used to specify the permissions that particular participants have in the network and to protect the information included in the created transactions and assets.

(ii) The Data Layer relies on Tangle (https://blog.iota.org/the-tangle-an-illustrated-introduction-4d5eae6fe8d4) using the main net of the IOTA (https://www.iota.org/) network relying on its Masked Authenticated Message (https://github.com/iotaledger/MAM) (MAM) feature. It is important to remark that Tangle cannot be considered a blockchain technology, as it does not use blocks. Instead, transactions are directly included to the network by validating two previous transactions called Tips, creating a Directed Acyclic Graph (DAG). In the Data Layer, rather than having a private network as it is done in the business layer, the data interface is connected to the IOTA main net using MAM, which features private and encrypted channels between peers in spite of being a permissionless network. With this approach our solution benefits from the computing power already deployed as part of the IOTA main net while ensuring that the acquired data can only be read by the authorized participants. In addition, data transactions sent though the MAM can be read as they are attached to the network, while they are Tips, without the need of waiting for their validation. This happens as the data transactions do not include tokens (cryptocurrency), so a double spending cannot happen for this kind of transactions.

The result of having these two layers in the blockchain-based communication system is that each node of the marketplace integrates a node of a private Hyperledger deployment together with a node of the public IOTA main net. Most of the actions of a layer are performed on its own ledger. However, there are crossed relations that represent the joint point among both layers. On the one hand, the data interface uses the business ledger for knowing details of the acquired datasets whose data have to be requested through MAM, or for enforcing the access policy at receiving data requests. On the other hand, the business interface introduces communication details (identifier of MAM channels) on the agreement setup process. Figure 4 shows the particular implementation of the blockchain-based communication system.

4.2.1. Business Layer. The business layer is in charge of the traditional functionalities of a marketplace that is the management of offerings and the help in the establishment of agreements. Regarding the offerings, the business layer implements the commercialization part of the TM forum's offerings lifecycle, namely, Active, Launched, and Deprecated. However, the first one is treated as local status only affecting the interaction with the user, only reflecting on the distributed ledger the statuses of Launched and Deprecated. With respect to the establishment of agreements the states

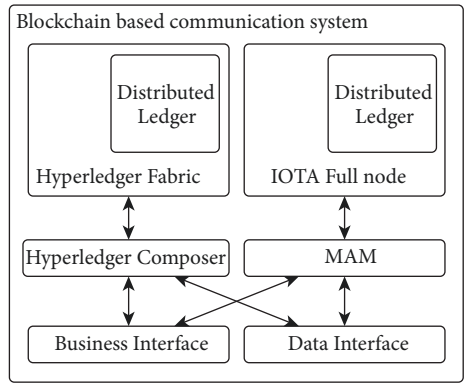

Figure 4: Blockchain-based communication system implementation.

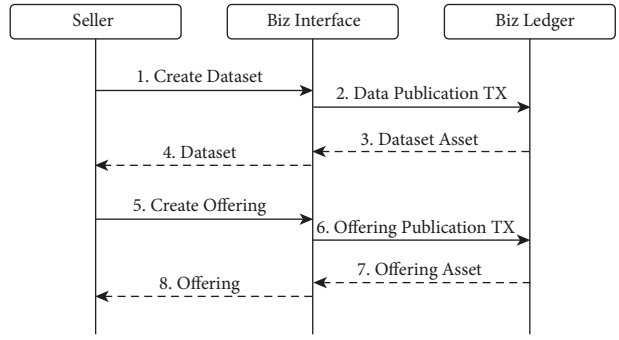

Figure 5: Offering publication.

Pending, Active, and Revoked are used to reflect whether an agreement is valid.

The Business Layer exposes the business interface to its users (API and Web) offering functionalities for creating offers, manage their life cycle including deprecation, defining the price models, and manage the creation and maintenance of agreements according to the different price models supported.

(1) Management of Offerings. The published information about advertised data is split into two different concepts, the description of the data with all the information required to identify a particular stream and the description of the business aspects related to its monetization and usage terms. Having these concepts uncoupled allows the definition of richer scenarios, including having the same data advertised in multiple offerings. For the publication of this information in the business ledger, the business interface generates two different types of transactions: the *Data Publication Transaction* and the *Offering Publication Transaction* as depicted in Figure 5.

Data Publication Transaction. The *Data Publication Transaction* exposes NGSI data managed by the *Context Broker* that the blockchain-based communication system is connected to, by listing exposed entities and their offered attributes. Entities are specified using values or patterns for their id

or any of their attribute values, in a query-like style. For the particular case of geolocalized entities, such filter can additionally include GeoJSON information used to filter the exposed entities. Examples of these definitions can be every entity within a given area, entities of a particular type, or entities whose name starts by a given string.

The attribute-based level of granularity supports the publication of a subset of attributes, keeping part of the entity private, thus not disturbing the existing processes and applications when data is monetized. Moreover, customers are allowed to choose which of the published attributes or entities they want to acquire, given that the price models support this situation.

The result of the consolidation of this transaction, that is the execution of the attached smart contract, is the creation of a dataset asset, being uniquely identified across the network, offered by the transaction issuer.

Offering Publication Transaction. The *Offering Publication Transaction* provides pricing and business details. Specifically, it holds a reference to the dataset identifier, the terms and conditions to be accepted by the customer on the acquisition, and the following monetization details:

(i) **Contracts**: they describe the duration of the agreement. The proposed solution deals with two kinds of contracts: (1) time-based contracts which specify the duration in time and (2) usage-based contracts which specify the acquired amount of data (e.g., 10000 entities).

(ii) **Characteristics**: they define selectable characteristics available to potential customers such as the query rate or subscription throttling.

(iii) **Pricing models**: they establish how and when the customers will be charged according to the chosen data, contract, and characteristics.

When more than one contract, characteristic values or price models are defined, customer can choose the desired set, and the final price adapts to such selection, as is explained in the price model section.

The smart contract of the transaction is in charge of verifying the existence of related dataset asset issued by the same user and generates in the *world state* a new offering asset.

Pricing Model. The pricing model used in the proposed solution is based on three different concepts: basic model, price alteration, and modifiers.

The **basic model** is the core of the pricing and establishes the period between charges, the information needed to compute them, and the basic price. In particular, the proposed solution supports the following:

(i) **One Time** payments which are charged once at acquisition time. For this model the (initial) price to be charged is specified.

(ii) **Recurring** payments which are charged periodically before the specified period. For these models, the (initial) price to be charged periodically is specified.

(iii) **Usage** payments which are charged periodically, at the end of the given period, and computed using the accounting information (usage) of the customer accessing to the data. For these models both the accounting unit and the (initial) price per accounting unit are specified.

Timing based contracts, namely, recurring or usage-based models, specify a contract duration in which the user can renew subscriptions. Under some conditions, the user may be forced to renew the subscription during the whole duration of the contract after each payment period (e.g., yearly contract with monthly payments). A new contract is to be established at the end of the contract duration, therefore allowing the update of the conditions.

The **price alterations** allow richer pricing models establishing how the final price should be increased or decreased according to certain conditions or time frame that are verified at charging time. Price alterations can be classified according to two different criteria: whether they are applied always or under certain conditions (e.g., the user has made more than 100 calls) or whether they are applied just at acquisition time or every time the customer is charged. Price alterations can be used, for example, for setting up an initial fee to a usage-based model or to include a discount when the customer makes more than a certain number of calls. Admission fee and usage discounts are examples of price alterations that might be applied, for example, to a basic subscription.

Modifiers are used to make the final adjustments to the charging price according to the different parameters selected by the customer when the offering was acquired. Modifiers use a weight based mechanism for the three types of modifiers:

(i) **Data attributes**: they define a weight between 0 and 1 for each published attribute, forcing a sum of 1. Therefore, acquiring all the published attributes for an entity does not change the price, while acquiring a subset makes a discount on it.

(ii) **Contract**: each of the available contracts is assigned a multiplying factor enabling us to increase or decrease the final price according to the selected contract (e.g., a 24-month subscription may be cheaper than a 12-month one).

(iii) **Characteristics**: each of the values of the included characteristics is assigned a multiplying factor, enabling us to increase or decrease the final price according to the selected values (e.g., a higher query rate may be more expensive)

(2) Management of Agreements

Agreement Setup Process. The Business Layer manages the distributed set up of agreements between sellers and customers in order to acquire access to the published data, as depicted in Figure 6. It is worth remarking that the data delivery performed as a result of setting up an agreement between two peers of the network is done through IOTA using private MAM channels, where only a publisher can send

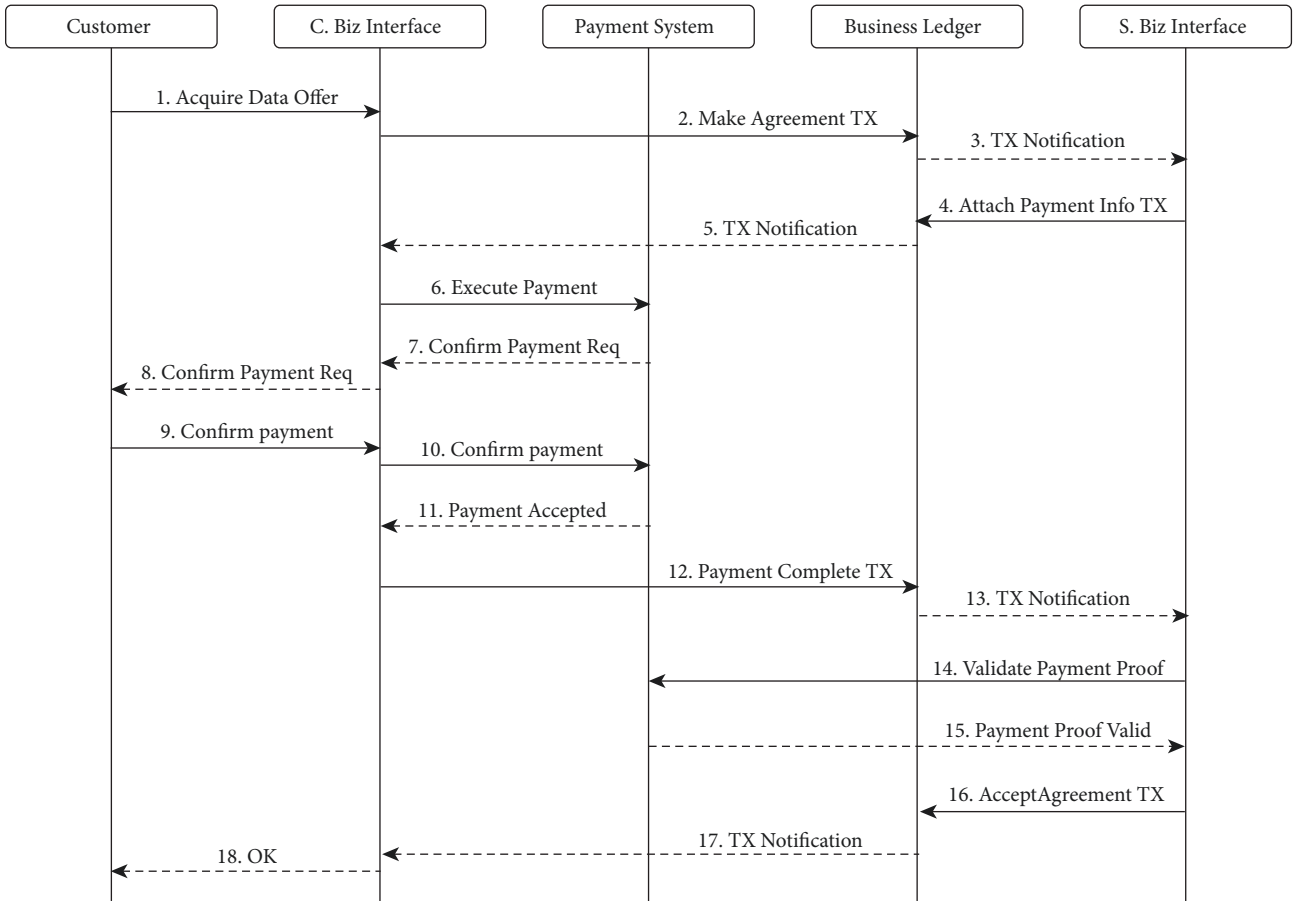

Figure 6: Agreement setup process.

transactions. In this regard, for the proposed solution two channels are required per agreement, one for the customers to publish their data queries and other for the sellers to submit the data. In addition, both peers need to know the *root ID* (address of the first message to be sent) and the encryption key for accessing a channel data. This information is distributed as part of the agreement setup as described in the following paragraphs.

The *Make Agreement Transaction*, run upon the acquisition request made by the customer, triggers the agreement setup and includes the chosen parameters, implying an implicit acceptance of the terms and conditions. In particular, this transaction includes the ID of the offering asset, the selected data attributes, the chosen pricing model, the selected contract, and the selected characteristics. The smart contract of this transaction computes the initial charge, unless the selected pricing is a usage model without initial fee, and creates the agreement asset. This asset is created on *pending* state meaning that the customer does not have access to the data until the pending payment is satisfied.

In addition, as part of the *Make Agreement Transaction* the customer business interface includes the MAM root ID and encryption key to be used for sending queries related to the ongoing agreement. This information is obtained from the data layer through the MAM interface and is sent encrypted using the public key of the data seller. This is the only information not accessible by the smart contract, since it is not needed for any validation.

Once the agreement is created the customer has to pay the pending amount. The proposed architecture does not impose any restriction about the payment method, intended to support PayPal, fiat or cryptocurrency transactions. Instead, it defines a pair of transactions used by the customer and seller, to verify external economic interchanges. In particular, *Attach Payment Info Transaction* is used by the seller node in order to include in the agreement asset the needed payment information that must be used by the customer node to perform the payment. Next, *Payment Complete Transaction* must be provided by the customer business interface once having paid, including the payment proofs. Note that the particular nature of the proof will depend on the payment method used.

Finally, the business interface of the data seller must issue an *Agreement Accepted Transaction* once finishing validating the payment proofs. The smart contract of this transaction changes the asset state to 'Active' and sets up the timestamp until which the agreement is valid. Additionally, this transaction includes the answer channel details, that is the MAM root ID and encryption key, both encrypted with the public key of the customer. In the case of not requiring initial

payment, payment verification transactions are not used, but this one remains due to the need of establishing the MAM answer channel.

Once the agreement has been created, the business interface of the customer node uses dataset asset details in order to create a NGSI registration in the local Context Broker. This registration specifies the data interface of the blockchain-based communication system as the data source enabling the automatic query forwarding.

It is noteworthy to mention that the seller does not directly interact in this process since it is performed automatically by her business interface after her having configured the payment preferences.

Agreement Settlement Process. When using a recurring or usage-based model, the validity period of the agreement is the payment period, so at the end of it, customer is in charge of renovating the agreement and sending a new payment according to the price model. For doing so, a new transaction called *Settle Agreement Transaction* has been defined, which in combination with *Payment Complete* and *Accept Agreement Transactions* are used to set up a new validity period, satisfy the payment conditions, and renew the MAM channels' credentials.

The *Settle Agreement Transaction* smart contract calculates the amount the customer has to pay for the particular settlement. In this regard, for recurring models it calculates the amount the customer has to pay for renewing a new period, while for usage-based-models, the smart contract applies the pricing model to the accounting information, whose gathering process is described in the Data Layer section. Finally, this transaction includes a flag used to specify whether the agreement is going to be renewed for new validity period. However, if customer is not willing to renew a recurring model and there is not any payment pending, this transaction is not needed since agreements are automatically invalid when the validity period ends.

If the contract duration is over, the customer will not be able to renew the agreement, so a new one is to be created with a *Make Agreement Transaction*.

Note that the proposed architecture is intended to be able to monitor the agreements and the payments, ensuring the nonrepudiation of the different interactions. The platform is not intended to make the effective enforcement of the agreement, but providing the tools to both customers and sellers to demonstrate without doubts that the agreements are satisfied. The actions that have to be performed when there is a violation of the agreements are out of the scope of this paper.

4.2.2. Data Layer. The Data Layer is in charge of performing the data delivery, enforcing access rights, and accounting for the data consumption in order to support usage-based pricing models and allow auditing customer service usage (e.g., for SLA enforcement). Data access is performed within the data ledger, therefore generating immutable trusted usage accounting, just like the business layer assures offerings and agreements.

The Data Layer is transparent to data applications deployed in the organization node, since every interaction made by such apps to consume context information is directly performed against the *Context Broker*. This broker leverages its federation mechanism, along with the NGSI Registrations created during the agreement setup, to retrieve remote (acquired) context data, as can be seen in Figure 7. The broker forwards queries and subscriptions received from the data apps to the data interface, which encodes them in the data ledger in the form of transactions for obtaining the data. Figure 8 shows the data delivery process for both data **queries** and data **subscriptions**.

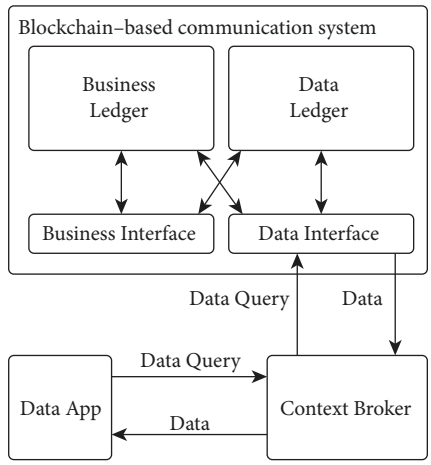

Figure 7: Data delivery architecture.

It is important to remark that the IOTA network is a permissionless distributed ledger, not allowing us to identify a particular participant of the peer-to-peer network as defined in our solution. To overcome this issue, the proposed solution uses the credentials (certificate and keys) of the business ledger to sign and hash-MAC all the messages included in the different MAM channels.

Queries. The delivery process for queried data, as depicted in Figure 8, starts with a Data App making a query to the Context Broker. This component uses the NGSI registration information to forward the query to the data layer of the blockchain-based communication system, whose data interface encodes the query as IOTA MAM message, the *Query Transaction*, in order to attach it in the customer MAM channel. The *Query Transaction* includes the agreement ID and all the information provided within the data query, including requested entities, attributes, filters, geographic area, etc.

Seller data interface is notified when the *Query Transaction* is attached to, which then checks the existence of the specified agreement at the business ledger and validates the queried data and the validity period through the Hyperledger Composer interface. If found, the requester is authorized to read the data, so the data interface performs the original query in the Seller Context Broker and encodes the result as a MAM message, the *Data Transaction*, which is then attached to the seller channel. The data interface of the customer receives a notification for the included *Data Transaction*, so it extracts the data and forwards it to the customer Context Broker.

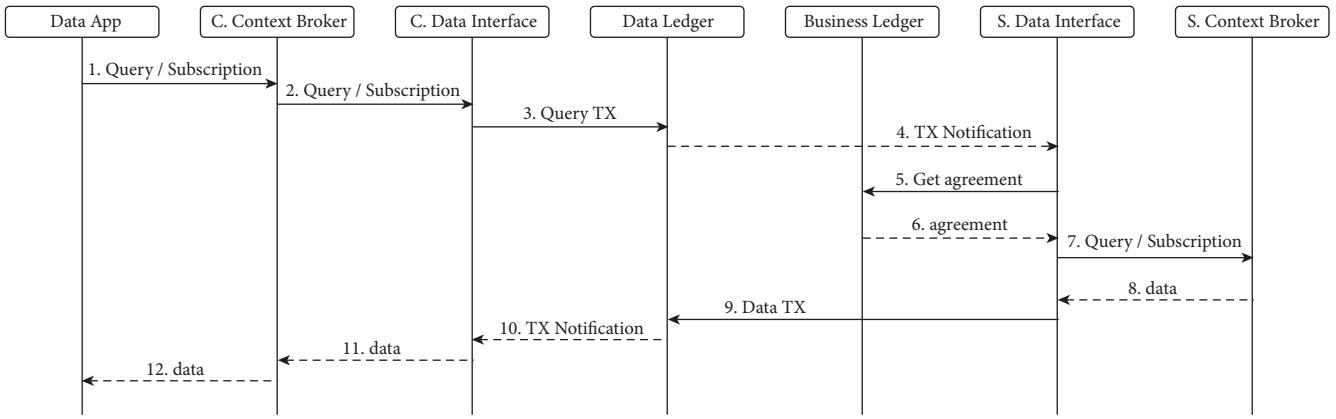

Figure 8: Data delivery process.

Subscriptions. The delivery process for subscribed data starts in a similar way as the queried data. A Data App or the CEP willing to process a right-time data stream creates a subscription in the local Context Broker. If such subscription is part of a NGSI registration, the Context Broker gets subscribed to the data source specified in the registration, which is indeed the data interface of the blockchain-based communication system. This component encodes the subscription as an IOTA MAM message in the customer channel (*Query Transaction*), which is received by the seller data interface. It checks whether the customer can subscribe to the particular data by querying for a valid agreement in Hyperledger and, if the customer is authorized, the data interface of the seller gets subscribed to the requested data in the Seller Context Broker, using the validity period of the agreement for fine tuning the duration of the subscription.

With this approach a subscriptions chain is created, so each time the subscribed entities are updated, the Seller Context Broker sends an asynchronous notification with the requested entities and attributes to the endpoint specified in the subscription. Such endpoint is the seller data interface, which encodes the notification as a *Data Transaction* within the seller MAM channel in the IOTA network. When the Data Transaction is read by the data interface of the customer node, the notification is extracted and forwarded to the customer Context Broker which sends it to the subscriber app. Note that in this process, the customer creates a single subscription, and then, a regular data stream is created from the seller to the customer.

Accounting. The proposed approach saves all the requests, subscriptions, and data responses as part of the IOTA network, making it possible to validate how the acquired service is being provided. However, in order to support the usage-based pricing models and the charging calculation performed by the smart contract of the *Settle Agreement* transaction, aggregated accounting information needs to be saved in the business ledger.

The accounting aggregation process is launched by the seller data interface, which retrieves all the *Query* and *Data Transactions* for a particular agreement on the data layer and generates an *Attach Accounting Transaction* in the business layer. This transaction includes the ID of the agreement, the timestamps of the accounted period, and the aggregated information which is useful for the usage-based models, including number of queries, number of downloaded entities, total bytes downloaded.

The validation of the *Attach Accounting Transaction* in the business ledger generates an Accounting Asset in "Pending" state, which needs to be confirmed by the customer. Upon its reception, the customer node checks the accounting information received against the data ledger and, if it is correct, it answers with an *Accounting Verified Transaction* in the business ledger referencing the pending Accounting Asset. As a result of this transaction the state of the Accounting Asset is set to "Verified".

The smart contract of the *Settle Agreement Transaction* uses the "Verified" Accounting assets in order to calculate the fee on usage-based pricing model agreements, therefore evolving the Accounting Asset state to "Charged".

5. Validation of the Proposal

This section analyses the feasibility of the proposed solution thought the case study described in Section 3 by describing how the required distributed marketplace among the smart building and the sensor company can be implemented according to the depicted architecture. In addition, it demonstrates how the usage of FIWARE technologies and FIWARE NGSI in a fog computing distributed scenario, where date is stored and processed locally, provides seamless interoperability between the deployed nodes. Figure 9 depicts the architecture of how the use case is implemented according to the proposed solution.

State Prior to Data Sharing. In an initial step, both nodes involved in the case study use a FIWARE deployment for the management of their internal data and IoT devices. In particular, the air quality node manages *AirQualityObserved* (http://fiware-datamodels.readthedocs.io/en/latest/Environment/AirQualityObserved/doc/spec/index.html) NGSI entities, which are created by its deployed IoT Agent from the raw IoT data of the node sensors. Additionally, a CEP is used to detect anomalous situations generating alerts

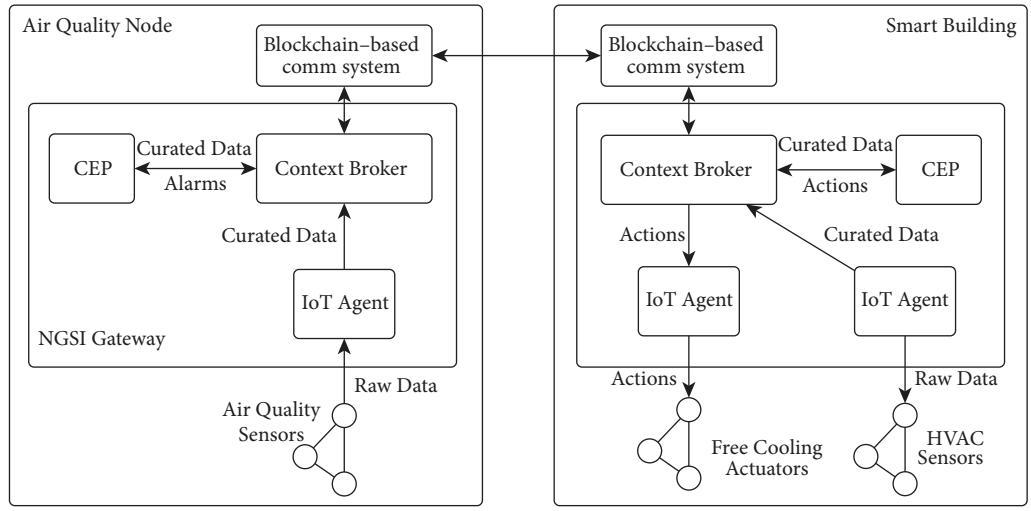

Figure 9: Architecture of the use case.

(http://fiware-datamodels.readthedocs.io/en/latest/Alert/doc/spec/index.html#alert-data-model). The *AirQualityObserved* entities include the timestamp and the location of the observation, several weather information, and the value of a set of pollutants. An example of an *AirQualityObserved* entity is shown below:

```
{
    "id": "Madrid-AmbientObserved-01",
    "type": "AirQualityObserved",
    "address": {
        "addressCountry": "ES",
        "addressLocality": "Madrid",
        "streetAddress":
        "Plaza de España"
    },
    "dateObserved":
    "2016-03-15T11:00:00",
    "location": {
        "type": "Point",
        "coordinates":
        [-3.712247222222222,
        40.423852777777775]
    },
    "precipitation": 0,
    "relativeHumidity": 0.54,
    "temperature": 12.2,
    "windDirection": 186,
    "windSpeed": 0.64,
    "airQualityLevel": "moderate",
    "reliability": 0.9,
    "CO": 500,
    "NO": 45,
    "NO2": 69,
    "NOx": 139,
    "SO2": 11,
    "CO_Level": "good",
    "NO_Level": "moderate"
}
```

On the other hand, the smart building manages NGSI information about the current status of its HVAC system (which incorporates free cooling) including temperature, humidity, water flows, pump speeds, and fan speeds, curated by its IoT agents. The CEP processes the NGSI data stream generating actions by creating *BuidingOperation* (http://fiware-datamodels.readthedocs.io/en/latest/Building/BuildingOperation/doc/spec/index.html) NGSI entities in the Context Broker. The IoT agents, subscribed to *BuildingOperation* entities, use this information to actuate over the IoT network.

Data Publication. With the operation of the different nodes setup, the owner of the air quality node wishes to publish some of the curated right-time data following the process defined in Figure 5. Every interaction mentioned in this section is based on such diagram. For this particular scenario, the values of NO_2, NO, CO, and SO_2 levels as well as temperature for a given area are advertised by invoking the business interface API (interaction 1 of the diagram).

```
POST https://bizinterface/api/datasets
{
    "id": "mad-air-quality-1",
    "description": "Temperature and NO2,
    NO, CO and SO2 levels in Madrid

Centre Area",
    "dataProvided":{
```

```
        "entities": [
          {
           "type": "AirQualityObserved"
          }
        ],
        "attrs": [
            "NO2", "NO", "CO", "SO2",
            "temperature"
        ]
    },
    "location": {
        "type": "polygon",
        "coordinates": [
          [-3.714248, 40.423035],
          [-3.711770, 40.424423],
          [-3.710300, 40.425142],
          [-3.709066, 40.424292],
          [-3.709302, 40.420707],
          [-3.714248, 40.423035]
        ]
    }
}
```

To monetize the dataset, the seller creates an offering using the business interface API (interaction 5 of the diagram). In this example, the offering includes eligible 12- or 24-month contracts where the former gives a 10% discount on the pricing. The pricing model is based in IOTA cryptocurrency (expressed in 1 k or 1 M units) and defines a usage-based pricing model of 1 KIOTA per downloaded entity with an initial fee of 1 MIOTA. In addition, the pricing includes a 10% discount when the customer downloads more than 10000 entities within a charging period of one month. An excerpt of the offering creation request can be found below, excluding some of the options for brevity:

```
POST https://bizinterface/api/offerings
{
    "id": "mad-air-quality",
    "dataset": "mad-air-quality-1",
    "liveCycleStatus": "Active",
    "terms": {
        "title": "Non-Commercial Use",
        "text": "The Offered data cannot
        be used with commercial purposes
or be reselled"
    }
     ...
    "contracts": [{
        "id": "2y"
        "type": "usage",
        "value": 24,
        "unit": "month"
    },
    ...
    ],
    "pricing": {
        "plans": [{
        "id": "usage",
        "type": "usage",
        "value": "1"
        "currency": "KIOTA",
        "unit": "entity",
        "alterations":[{
        "description":
        "Initial fee of 1 MIOTA",
        "type": "fee",
        "isRecurring": false,
        "value": "1000",
        "isPercentage": false
        "condition": ""
        },...]
        }],
        "modifiers": {
        ...
        "attributes": {
        ...
        "temperature": 0.1
        },
        "contracts":[{
        "id": "1y",
        "weight": 1
        },{
        "id": "2y",
        "weight": 0.9
        }]
        }
    }
}
```

Upon each seller request, the business interface attaches a transaction in Hyperledger Composer (interactions 2 and 6 of the diagram). These two transactions are introduced in the ledger signed by the seller, keeping the offering conditions immutable and public. An example of *Offering Publication Transaction*, where *offering* includes the monetizing information, can be found below.

```
{
    "$class": "org.conwet.biznet.
    OfferingPublication",
    "offering": {
    ...
    },
    "description": "This offering
    includes access to air quality data",
    "liveCycleStatus": "Active",
```

```
    "dataset": "resource:org.conwet.
    biznet.Dataset#mad-air-quality-1",
    "owner": "resource:org.conwet.
    biznet.User#seller-1",
    "transactionId": "ef90f367684fbf57
    324822cb383618d5",
    "timestamp": "2018-06-04T10:01:24.
    582Z"

}
```

Data Acquisition. Offerings and datasets are maintained within the network as Hyperledger Composer assets, snapshotted in the Fabric world state, and can be discovered using the search API provided by the business interface. Once a suitable offering has been found, and its details are obtained, it can be acquired following the process detailed in the diagram of Figure 6. Every interaction mentioned in this section is based on such diagram.

```
GET https://bizinterface/api/offerings?
filter=
[{

    "id": "mad-air-quality",
    "dataset": "mad-air-quality-1",
    "liveCycleStatus": "Active",

...
}]
GET https://bizinterface/api/datasets/
mad-air-quality-1
{

    "id": "mad-air-quality-1",
    "description": "Temperature and NO2,
    NO, CO and SO2 levels in Madrid

Centre Area",
...
}
```

In the current case study, the smart building owner acquires access to the air quality data stream so it can be incorporated to the free cooling management, irrespectively of later access through queries or subscriptions. From the air quality offering she selects the 24-month contract and all the advertised pollutants (NO_2, NO, CO, and SO_2) but not the temperature. The process is initiated with the acquisition request made to the business interface (interaction 1 of the diagram).

```
POST https://bizinterface/api/offerings/
mad-air-quality/acquisitions
{

    "attributes": ["NO2", "NO", "CO",
    "SO2"],
    "characteristics": [] ,
    "contract": "2y",
    "pricing": "usage"

}
```

This acquisition request generates a *Make Agreement Transaction* (interaction 2) whose smart contract calculates the initial fee, which is then included within the created agreement asset. The pricing model of the air quality data offering includes an initial fee of 1 MIOTA which in combination with the multiply factor of the chosen contract and the weights of the selected attributes results in a pending payment of 810 KIOTA (1000 KIOTA $*$ 0.9 $*$ 0.9 = 810 KIOTA).

Since the current agreement includes a pending payment, the business interface of the seller submits an *Attach Payment Info* Transaction, including its IOTA addresses for receiving the payment (interaction 4).

```
{

    "$class": "org.conwet.biznet.
    AttachPaymentInfo",
    "method": "IOTA",
    "paymentInfo":
    "SJEDHICBWBRGB9XDHCOL...
    DEYOWVOSUPGIBBUALNGDTSJ9A",
    "agreement": "resource:org.conwet.
    biznet.
    Agreement#agreement-1",
    "owner": "resource:org.conwet.
    biznet.
    User#seller-1",
    "transactionId":
    "ef90f5696c4f4fff324d22cb322618d5",
    "timestamp": "2018-06-04T14:25:17.
    362Z"

}
```

After the execution of the agreement setup process, the created agreement asset is in *Active* state, including the MAM info for customer and seller channels, and the timestamp with a validity period of one month (the default for usage models). The world state contains such active status backed up by the immutable transactions (interactions 2, 4, 12, and 16) and relate to the original offering which cannot be modified. It can be found below the resulting agreement asset after the process

```
{

    "$class": "org.conwet.biznet.
    Agreement",
    "agreementId": "agreement-1",
    "offering": "resource:org.conwet.
    biznet.
```

```
    Offering#mad-air-quality",
    "customer": "resource:org.conwet.
    biznet.
    User#customer-1",
    "options": {
        "attributes":
         ["NO2", "NO", "CO", "SO2"],
        "characteristics": [] ,
        "contract": "2y",
        "pricing": "usage"
    } ,
    "payment": {
        "amount": "810",
        "currency": "KIOTA"
        "method": "IOTA",
        "SJEDHICBWBRGB9XDHCOL...
        DEYOWVOSUPGIBBUALNGDTSJ9A"
    },
    "channels": {
        "query": "...",
        "data": "..."
    }
    "liveCycleStatus": "Active",
    "validTo": "2018-07-04T14:32:19.
    574Z"

}
```

Once the acquisition has finished, the business interface creates the NGSI registration for the pollutant information of the air quality data served by the seller, which is used by the Context Broker in order to forward queries and subscriptions. An NGSI registration example is written below:

```
POST https://customerbroker/v2/
registrations
Fiware-Service: seller-1
{
    "description": "",
    "dataProvided": {
        "entities": [ {
          "type": "AirQualityObserved"
        } ],
        "attrs": [
          "NO2", "NO", "CO", "SO2"
        ],
        "expression": "georel=coveredBy&
        geometry=polygon&coords=

-3.714248,40.4..."

    },
    "provider": {
        "http": {
          "url":
          "https://datainterface/"
        }
    }

}
```

Data Consumption. Within the smart building node, the processing of the air quality data for actuating in the free cooling system is done in the CEP. This component searches for patterns in data stream, which in this scenario implies detecting high pollutant levels during a given period of time. To set up the air quality data stream feeding the CEP, the process described in Figure 8 is started by creating the following subscription in the local Context Broker (interaction 1).

```
POST https://customerbroker/v2/
subscriptions
Fiware-Service: seller-1
{
    "description": "",
    "subject": {
      "entities": [ {
       "idPattern": ".*",
       "type": "AirQualityObserved"
       } ],
      "condition": {
       "attrs": ["NO2", "NO", "CO",
       "SO2"]
       }
    },
    "notification": {
      "http": {
       "url":
     "http://cep/notifications/"
      },
      "attrs": ["NO2", "NO", "CO", "SO2"]
    }

}
```

As a consequence, a subscriptions chain is created propagating this subscription through the blockchain-based communication system, permanently storing the *Query Transaction* used for the propagation (interactions 2 to 7).

Each time the value of a pollutant of the subscribed entities changes, a notification flows from the seller to the customer, as described in interactions 8 to 12 of the diagram, through the seller MAM channel as a *Data Transaction*. Bellow it can be found an example of *Data Transaction* (currency-less IOTA transaction with the MAM message encoded in the field *signatureMessageFragment* as trytes).

```
{
```

```
    "hash": "IKCCXOBQTB9ORP9...
    VU9BGURP9TA9999",
    "signatureMessageFragment":
    "AHBAUZAFWNT...FDLIXXKPDVO",
    "address": "AQKWEZGRHHTWXGUUQD9...
    VORZVV9BWAULDEEPCEUL",
    "timestamp": 1522573490,
    "value": 0,
    ...
    "nonce":
    "DWGVRWWJYNHJJFIEKOXNOCFBDPC"

}
```


The field *signatureMessageFragment* in such currency-less IOTA transaction represents the actual MAM message. The most relevant information of the content of such field is decoded below:


```
{

    "state": {
      "subscribed":[],
      "channel": {
         "side_key": "ADMDGDAMDCFDTCHD",
         "mode": "restricted",
         "next_root":
         "L9XMV9C9WGDCBNE...
         BSHPMBEVXKDUB",
         ...
       },
      "seed": "HEBZKASGWWTWWP...
      DMVJDSMFOFMUCGBX"
    },
    "payload": "trytes(encrypted
    (encryption-key, {
      "subscriptionId": "12345",
      "data": [
       {
        "id": "Madrid-Ambient
        Observed-01",
        "type": "AirQualityObserved",
        "CO": 500,
        ...
       }
      ],
      "hmac": "..."

  ))",
"root": "DAYKXMJLMAEKMORPCZA...
CVGEGLNEOONWGKOHVGE",
"address": "AQKWEZGRHHTWXGUUQD9...
VORZVV9BWAULDEEPCEUL"
}
```


In this scenario, the air quality data is offered under a usage pricing model. In this regard, in each charging period (monthly) the information available in both seller and customer channels (*Query and Data Transactions*) is aggregated in an *Attach Accounting Transaction* which is submitted to the business ledger by the seller and validated by the customer, having access to data ledger channels and their immutable data sharing transactions acting as accounting records.


```
{

    "$class": "org.conwet.biznet.
    AttachAccounting",
    "agreement": "resource:org.conwet.
    biznet.Agreement#agreement-1",
    "start": "2018-06-04T14:25:17.362Z",
    "end": "2018-07-04T14:32:19.574Z",
    "accounting": {
      "entities": "20000",
      "mb": "2.27",
      "calls": "1500"
    },
    "transactionId": "96acb5696c4f4
    fff328cd6cb322618d5",
    "timestamp": "2018-07-04T14:32:19.
    574Z"

}
```


During the settlement of the agreement, the submitted accounting is used for feeding the agreement pricing models. In this scenario, assuming the smart building has downloaded 20000 entities, the pending payment would be 18 MIOTA (20000 entities $*$ 1 KIOTA/entity = 20 MIOTA with the 10% discount for downloading more than 10000 entities = 18 MIOTA). This would be calculated by the smart contract of the *Settle Agreement Transaction*.

6. Conclusions

Distributed marketplaces with peer-to-peer data delivery models are more suitable than centralized approaches for monetizing data in fog scenarios, where produced data is preferred to be stored and processed locally. This paper presents such a peer-to-peer distributed data marketplace with advanced business capabilities, where trust is assured by the usage of blockchain technology. The solution combines different distributed ledgers in order to satisfy the requirements of each layer.

Unlike other marketplace solutions using blockchain technology that only focuses on implementing aggregation and search functionality of data and offerings, and assisting in the agreement process, our solution goes a step further and also allows to register faithful and verifiable accounting records used on usage models of pricing when performing

the data distribution and access control in a peer-to-peer data marketplace.

Moreover, the proposed distributed marketplace is compatible with advanced information distribution schemas like the FIWARE architecture of federated Context Brokers. In this scenario, query- or subscription-based access to acquired data is performed transparently for the data applications running in the fog node, which access the local Context Broker for getting the data without knowing whether the latter is local or remote (i.e., acquired from a different node of the network).

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The research leading to these results was partially funded by the European Commission Horizon 2020 Research Framework Programme FI-NEXT Project under Grant agreements ID 732851. This article expresses the opinions of the authors and not necessarily those of the European Commission. The European Commission is not liable for any use that may be made of the information contained in this paper.

References

[1] The Economist, "The worlds most valuable resource is no longer oil," https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data.

[2] N. Kroes, "Digital Agenda and Open Data," http://europa.eu/rapid/press-release_SPEECH-12-149_en.htm.

[3] T. W. Malone, J. Yates, and R. I. Benjamin, "Electronic markets and electronic hierarchies," *Communications of the ACM*, vol. 30, no. 6, pp. 484–497, 1987.

[4] J. Y. Bakos, "A strategic analysis of electronic marketplaces," *MIS Quarterly: Management Information Systems*, vol. 15, no. 3, pp. 295–310, 1991.

[5] C. C. Albrecht, D. L. Dean, and J. V. Hansen, "Marketplace and technology standards for B2B e-commerce: progress, challenges, and the state of the art," *Information & Management*, vol. 42, no. 6, pp. 865–875, 2005.

[6] P. Brody and V. Pureswaran, "Device democracy: Saving the future of the Internet of Things IBM," *IBM*, 2015.

[7] B. Varghese, N. Wang, S. Barbhuiya, P. Kilpatrick, and D. S. Nikolopoulos, "Challenges and Opportunities in Edge Computing," in *Proceedings of the IEEE International Conference on Smart Cloud, SmartCloud '16*, pp. 20–26, USA, November 2016.

[8] E. (O'Reilly) Dumbill, "Data markets compared - A look at data market offerings from four providers , OReilly Radar," *E. (O'Reilly) Dumbill*, 2012, http://radar.oreilly.com/2012/03/data-markets-survey.html.

[9] S. Mansfield-Devine, "Beyond Bitcoin: using blockchain technology to provide assurance in the commercial world," *Computer Fraud and Security*, vol. 2017, no. 5, pp. 14–18, 2017.

[10] V. Gramoli, "From blockchain consensus back to Byzantine consensus," *Future Generation Computer Systems*, 2017.

[11] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," in *Proceedings of the IEEE Security and Privacy Workshops, SPW '15*, pp. 180–184, IEEE, May 2015.

[12] P. B. Nichol and J. Brandt, "Co-Creation of Trust for Healthcare: The Cryptocitizen Framework for Interoperability with Blockchain," *ResearchGate*, 9 pages, 2016.

[13] J. Benet, "IPFS - Content Addressed, Versioned," *P2P File System , CoRR*, vol. 3, no. Draft 3, 2014.

[14] A. Azaria, A. Ekblaw, T. Vieira, and A. Lippman, "MedRec: Using blockchain for medical data access and permission management," in *Proceedings of the 2nd International Conference on Open and Big Data, OBD '16*, pp. 25–30, Austria, August 2016.

[15] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, and M. Guizani, "MeDShare: Trust-Less Medical Data Sharing among Cloud Service Providers via Blockchain," *IEEE Access*, vol. 5, pp. 14757–14767, 2017.

[16] A. Ouaddah, A. Abou Elkalam, and A. Ait Ouahman, "FairAccess: a new Blockchain-based access control framework for the Internet of Things," *Security and Communication Networks*, vol. 9, no. 18, pp. 5943–5964, 2017.

[17] K. Mišura and M. Žagar, "Data marketplace for Internet of Things," in *Proceedings of the 1st IEEE International Conference on Smart Systems and Technologies, SST 2016*, pp. 255–260, Croatia, October 2016.

[18] C. Perera, S. Y. L. Wakenshaw, T. Baarslag et al., "Valorising the IoT Databox: creating value for everyone," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 1, 2017.

[19] D. Wörner and T. Von Bomhard, "When your sensor earns money: Exchanging data for cash with Bitcoin," in *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp '14*, pp. 295–298, ACM, September 2014.

[20] K. Noyen, D. Volland, D. Wörner, and E. Fleisch, *When Money Learns to Fly: Towards Sensing as a Service Applications Using Bitcoin*, 2014.

[21] J. McKane, "Bitcoin vs Ethereum – What to expect in 2018," https://mybroadband.co.za/news/cryptocurrency/-bitcoin-vs-ethereum-what-to-expect-in-2018.html.

[22] Ocean Protocol, "Ocean Protocol: A Decentralized Substrate for AI Data & Services - Technical Whitepaper," Tech. Rep., 2018.