future internet

MDPI

Article

Enhancing IoT Data Dependability through a Blockchain Mirror Model

Alessandro Bellini [1,*], Emanuele Bellini [1,2], Monica Gherardelli [3] and Franco Pirri [3]

[1] Mathema s.r.l., 50142 Florence, Italy; emanuele.bellini@mathema.com
[2] Center of Cyber Physical Systems, Khalifa University of Science, Technology & Research, Abu Dhabi 127788, UAE
[3] Department of Information Engineering, University of Florence, 50139 Florence, Italy; monica.gherardelli@unifi.it (M.G.); fpirri@gmail.com (F.P.)
* Correspondence: abel@mathema.com; Tel.: +39-346-646-4647

Received: 28 March 2019; Accepted: 15 May 2019; Published: 21 May 2019

Abstract: The Internet of Things (IoT) is a remarkable data producer and these data may be used to prevent or detect security vulnerabilities and increase productivity by the adoption of statistical and Artificial Intelligence (AI) techniques. However, these desirable benefits are gained if data from IoT networks are dependable—this is where blockchain comes into play. In fact, through blockchain, critical IoT data may be trusted, i.e., considered valid for any subsequent processing. A simple formal model named "the Mirror Model" is proposed to connect IoT data organized in traditional models to assets of trust in a blockchain. The Mirror Model sets some formal conditions to produce trusted data that remain trusted over time. A possible practical implementation of an application programming interface (API) is proposed, which keeps the data and the trust model in synch. Finally, it is noted that the Mirror Model enforces a top-down approach from reality to implementation instead of going the opposite way as it is now the practice when referring to blockchain and the IoT.

Keywords: IoT; blockchain; IoT security; IoT data dependability; Mirror Model; representation and trust models; API gateway

1. Introduction

One subtle aspect of the alert recently raised by Dr. Geneveva Allen about a "science crisis" [1] is the observation that current machine learning algorithms may discover patterns in data that exist only in data but not in the real world. This is particularly true of data obtained by external sources where a validation process is barely exposed or accomplished. The emphasis here is not in external sources but in the lack of a credible validation process of data. This is "a fortiori" true in the case of data generated by the Internet of Things (IoT). In fact, the IoT relies (and will rely) heavily on Artificial Intelligence (AI) techniques to provide security and reliability [2]. The infrastructure network of devices that forms the base of any IoT system is a remarkable data producer and AI is aimed at discovering patterns in these data that, in turn, may provide hints for detecting or preventing security breaches or provide working optimization or reliability (e.g., detecting those devices that are not working properly).

However, these desirable benefits are gained if the data produced by the IoT are dependable. Data from a sensor are a representation of the state of the world and should not be confused with reality. For instance, a thermometer could measure a body temperature of 40 °C while the real body temperature is 36 °C. This discrepancy could be caused, for example, by a defect in the thermometer or by an incorrect procedure of measuring. At the same time, we must have certainty that this particular thermometer applied to this particular subject at that particular time produced a measure of 40 °C. We have to be certain of the data values produced by the sensors (eventually not corresponding to a

faithful representation of the reality), i.e., that these data have been not modified or altered in any form. This is a fundamental requirement to accomplish computations or investigations that may amend errors and yield practical results. In other words, we need a mechanism to render the representation of data produced by IoT networks trustable. This is where blockchain comes into play [3–5]—providing data immutability and consensus among peers when data are updated.

The article is organized as follows: In Section 2 we present the state of the art related to the trust of data and the contribution of blockchain in the IoT domain. In Section 3 we propose a simple formal model named "the Mirror Model" where a measure (or a generic data) coming from an IoT network is associated (mirrored) with a corresponding registration in a blockchain network, thus providing dependability to data which, in turn, may be safely used in any subsequent AI pattern discovery or reasoning. In Section 4 we sketch a possible implementation of the Mirror Model, and in Section 5 we draw some short conclusions.

2. State of the Art

Trust is a well-recognized property of data and data exchanges [6], but the requirement of data trust has highly increased in relevance in recent years when large IoT networks began operating in different and critical domains (health, finance, military and similar) [7,8]. The intuitive concept of adding trust to a dataset that will be exchanged among several parties emerged several years ago, with a problem of stability and discoverability of the Internet resources over time (http 404 error). The issue has been addressed by implementing persistent identifiers (e.g., CoolURI, Digital Object Identifier, National Bibliography Number) and digital preservation services [9,10] in order to guarantee the trustworthiness of the digital resources exploited. The introduction of blockchain technology intuitively opens new technical possibilities to build trust for digital datasets that have been only partially explored in the current publications [11,12]. In particular, a formal argumentation of the precise meaning of adding trust to a dataset has not been provided yet. One of the reasons for this surprising omission may lay in the novelty of the blockchain technology (the technology aimed at providing trust) and its origin (Bitcoin). Technological issues are prominent in any review of blockchain and blockchain in the context of the IoT [13,14], and a more abstract and detached reflection on the conceptual meaning of trust is seldom considered. The fundamental step on the way toward more abstraction layers is the recognition of the separation of concerns between reality and representation and between representation and trust, as shown in this paper. The idea of separation of concerns through different layers of abstraction is well known in traditional areas such as database research, where it is common practice designing a conceptual model that, in turn, will be transformed in a database schema. Indeed, even in the area of blockchain this separation has been highlighted [15]. The substantial differences between database and distributed ledger technologies (and blockchain) are clear and proposals in the form of decision trees are also available to assist in the choice and use of both technologies. Nevertheless, solutions based on so-called "blockchain databases" [16], whose aim is the hybridization of databases and blockchain, are not so uncommon. This is not our position. We believe that databases pertain to the representation domain and blockchains to the trust domain, thus providing a clean and fruitful separation of concerns.

3. The Mirror Model

Before delving into the formal details of the Mirror Model, the context within which the model is supposed to work should be clear. In particular, our argumentation takes place in three correlated domains (Figure 1):

- Reality;
- Representation;
- Trust.

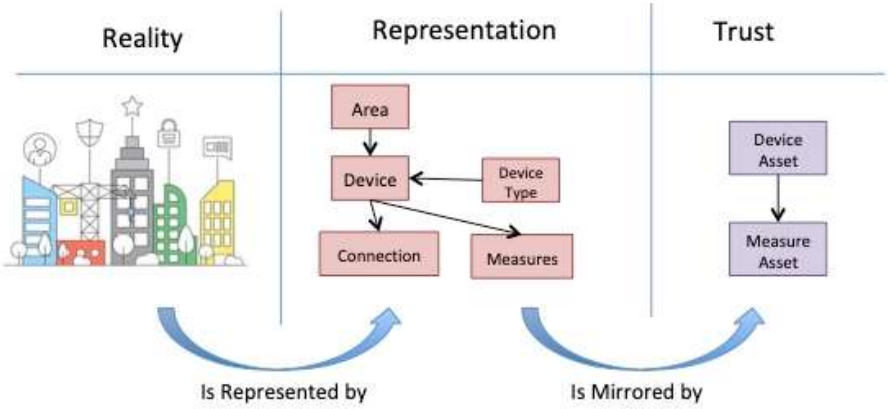


Figure 1. Mirror Model domains.

Reality is the domain where things actually exist and act. In the domain of Reality, we refer to entities, actions, reactions, events and so on. The Representation domain is the model of Reality we are interested in. In Representation, we refer to objects, relationships, attributes, functions, transformations and so on. The Trust domain is where objects in the Representation domain are made dependable and we refer to asset, transactions and immutable logs. In other words, Reality is where things objectively exist, independently of any efforts made by humans to design or control the Reality, Representation is a model conceived by architects or engineers or scientists to depict the aspects of interest or concern in the Reality and, finally, Trust is a structure of validation of the Representation of Reality whose aim is to make the Representation and its data dependable.

Trust in not the domain that makes Reality directly dependable, but it is the means to make the Representation directly dependable. Through the Trust and Representation domains we can act indirectly on the Reality to make it more reliable and secure for the benefit of humans and the environment in which they live. Trust is evidently the domain of blockchain for the simple reason that blockchain enforces the necessary mechanisms of immutability and consensus and makes available an immutable history of stored events. The Reality is connected to the Representation by the efforts of the designer while the Representation and Trust domains are connected by the Mirror Model.

Given this necessary introduction of the three basic domains, the Mirror Model may be defined through a simple formal notation [17] whose aim is to provide conciseness, rigor and precision. The intricacies of the IoT reality, the corresponding modeling and the very technical knowledge needed in blockchain may be synthesized in a discourse that highlights the essential principles, abstracting from the accidental details of implementation. The Mirror Model is subdivided in three parts:

- the Mirror, that sets the formal foundations to connect an object from the Representation domain to a corresponding asset in the Trust domain;
- the Horizontal Trust, that establishes the property that a collection of objects must possess in order to be trusted;
- the Vertical Trust, whose aim is to maintain trust in an object over time in the Representation domain.

3.1. The Mirror

A "mirror" is an asset in the Trust domain to be associated to an object in the Representation domain. The association between an object, o, and a "mirror", a, is accomplished as a function:

$$m : R \rightarrow T$$
$$m(o) \equiv a : T \cdot o.id = a.id \wedge$$

where $P(o,a)$ is a predicate, that is a function whose signature is:

$$P : T \times R \rightarrow \boldsymbol{B}$$

where ***B*** is the Boolean set whose values are True and False.

It is supposed that any object or asset is uniquely identified by an "*id*" that is expressed with the syntax *o.id* (*a.id* in the case of an asset).

The Trust domain, *T*, is immutable in that it complies with the following invariant:

$$\forall\, a : T \cdot G(a = a)$$

where *G* is the temporal operator "always" [18,19]. This invariant simply states the fact that *T* is the domain whose assets never change (assets are always equal to themselves), i.e., they are immutable. (The Trust domain, i.e., the blockchain domain, has many other properties besides the immutability of assets. For instance, in a blockchain an asset may be not modified without the consensus of endorsing parties, but this and other properties are not functional to the present argumentation and their formalization is beyond the scope of this paper.) An identifier is supposed to be unique within the whole domain; something that resembles a Uniform Resource Identifier (URI) or a Digital Object Identifier (DOI).

The mirror function, *m*(), given an object, *o*, in *R*, selects a corresponding and unique asset in *T*, where object and asset have the same "ids" and a generic predicate *P*() is satisfied for both the object and the asset. The function of the predicate in the mirror model is to add flexibility and to avoid objects and corresponding assets that are forced to share the same identical structure and contents. Of course, the trivial identity predicate stating $o = a$ is admitted and possible but, in general, objects and corresponding mirror assets possess different structure and contents.

Example: *Let consider an object that represents a real-world thermostat. Its structure could be as follows:*

Thermostat::=
id: N;
brand: String;
*manual: Bit **
status: {"on", "off"}

An instance of the thermostat could be:

thermo = {id 35, brand "Nest", manual <0011000 . . . >, status "on"}.

A corresponding valid asset could be:

a_thermo = {id 35, hash_manual <1753223>}.

In this case the structure of the asset could be:

AThermostat::=
id: N;
hash_manual: [0|1|..|9].*

Given the fact that the basic condition is satisfied, i.e., *thermo.id* = *a_thermo.id*, the validating predicate could be as follows:

P(o,a) = hash(o.manual) = a.hash_manual.

In other words, the asset stores the *hash()* of the manual. This is the practice in structuring the ledger of a blockchain, particularly when the original object is described using a large amount of information such as a document or an image.

3.2. The Horizontal Trust

Given a collection of objects, C, in *R*, i.e., $C \subseteq R$, *C* is said to be "horizontally trusted" by a "mirror set", *L*, in *T* (the ledger), $L \subseteq T$, when the following predicate applies:

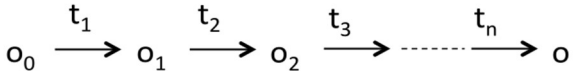
$$is\_h\_trusted:\ \boldsymbol{P}(R) \times \boldsymbol{P}(T) \rightarrow \boldsymbol{B}$$
$$is\_h\_trusted\ (C, L) \equiv$$
$$\forall\, o : C \bullet \exists \left| a : L \bullet a = m(o) \right.$$

In other words, a collection (or class) of objects in R is "horizontally trusted", if an only if it has a mirror counterpart in a ledger in T. The practical use of this property is for the validation of a current set of objects in R (i.e., the current model of reality).

The concept of horizontal trust guarantees that all entities in a model are trusted. If, for example, in the Representation model the entity Thermostat is present, then Thermostat is horizontally trusted if any single Thermostat instance (i.e., all thermostats in the model) is trusted. Of course, over time, objects undergo transformations as they happen to their counterparts in the Reality domain.

3.3. The Vertical Trust

Real-world entities are subjected to events that alter their states. Correspondingly, objects in the Representation domain have to undergo transformations that keep their states in synch with their real-world counterparts:



This chain of object transformations may be reduced to:

- o_0, an initial object state,
- $<t_1, t_2, t_3, \ldots, t_n>$, an ordered sequence of valid transformations where each transformation t_i is a member of:

$$Transf:\ R \rightarrow R$$

Each transformation is valid if it preserves the *o.id* in the application of the transformation. The chain of valid transformation could be rewritten as follows:

$$o = t_1(o_0); t_2; t_3; \ldots ; t_n.$$

It is now convenient to introduce a new function named *history():*

$$history:\ R \times R \times Transf^n \rightarrow R^n$$

$$history\ (o,\ init,\ t) \equiv \langle init, o_1, o_2, \ldots, o_i, \ldots, o_n = 0\rangle : R^n \bullet o_i = t_1(init); t_2; \ldots; t_i$$
$$i:\ [1..n].$$

history() is aimed at computing the sequence of the versions of a given object, o, starting from an initial version, *init*, and a sequence of transformation, t.

The property of "vertical trust" may now be conveniently defined as:

$$is\_v\_trusted:\ R \times R \times Transf^n \times P(T) \rightarrow B$$
$$is\_v\_trusted(o,\ init,\ t,\ L)\ \equiv \forall i : [1..n] \bullet \exists \left| a_i : L \bullet a_i = m(history(o, init, t)_i) \right.$$

In other words, for each version in the history of a given object, there exists a corresponding "mirror asset" in the ledger L, i.e., there is a corresponding history of mirror assets in L (Figure 2).

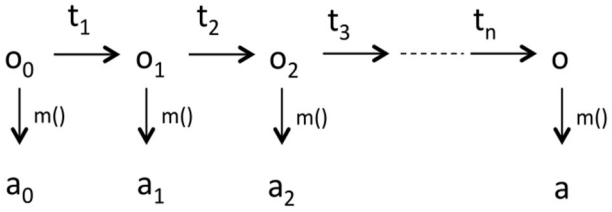


Figure 2. Mirror history.

The concept of vertical trust may be informally defined as the property of an object that is initially trusted and that keeps being trusted throughout its life cycle, that is in all of its state transitions. As a consequence, the concept of horizontal trust may be considered as the property of a set of objects to be currently trusted while vertical trust is the property of a single object to stay trusted over time. Of course, both properties are desirable in a model—we would like to have all objects trusted now and forever.

4. Practical Consequences

Blockchain may effectively provide trust to a model that represents some portion of the real world if the life cycles of prominent objects in the model are kept in synch with mirror assets in the blockchain. This has some practical consequences that we should note.

The first concerns time. In fact, it is highly recommended to mark objects and corresponding mirror assets with a timestamp. The ordinary object id serves to keep the identity of an object that corresponds to the uniqueness of the represented real-world entity and time serves to register the changes of state through the sequence of object versions (the object history). Moreover, time is a fundamental element of trust. In fact, the timestamp in the mirror object is an undeniable declaration that at that time the object identified by *o.id* (i.e., *a.id*) possessed exactly this mirror (i.e., an image of the original object mediated by the predicate, *P(o,a)*).

The second major practical consequence of the Mirror Model is that transformations of objects should be kept in synch with transactions of the corresponding mirror assets. This is the only way to ensure that objects will be trusted throughout their whole life cycle, i.e., objects that are dependable and can be processed to enforce security and reliability. To achieve this synchronization, it is necessary that whenever an object in the Representation model is engaged in a transformation, the corresponding mirror asset engages in a transaction (processed by a "smart contract") that keeps the new mirror asset in synch with the new version of the object after the transformation. This effect could be achieved through a mechanism of application programming interface (API) composition (Figure 3).

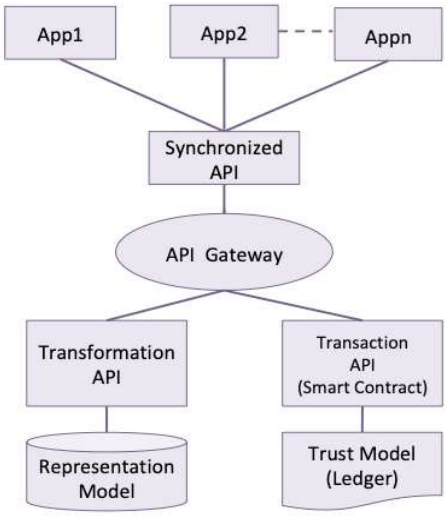


Figure 3. Synchronized API and API gateway.

It could be imaged to have two distinct sets of services structured in two APIs:

- The Transformation API implementing the services ordinarily used to upgrade the classic information system that model the IoT scenario;
- The Transaction API implementing the "smart contracts" used to upgrade the ledger in the blockchain.

These two APIs may be integrated and composed in one single synchronized API through the mediation of one of the many available API gateways [20]. In this way, applications (App_1, App_2, ... , App_n) refer to the service of a single API—the Synchronized API—that automatically keeps the Representation and Trust domains in synch, thus providing a trusted model.

This approach has the nice effect of allowing for a separate design, implementation and test of the transformation API (which may be informally named "API as usual", i.e., the ordinary web services organized in an API commonly implemented in a software platform) and the transaction API, that is the final product of "smart contracts" programming. In the end, both Transformation and Transaction APIs are composed in the final and unique Synchronized API. This decomposition of efforts is a direct consequence of the initial conceptual choice to structure the model in the three domains—Reality, Representation and Trust—and it is also a proof of how abstraction may lead to practical benefits.

These practical suggestions that stem directly from the theoretical Mirror Model may be applied in any practical application. We have indeed followed these design lines in the realization of a system aimed at providing Utilities Meter Data Collection and Management (UMDCM). The complete description of this case application is beyond the scope of this paper and it could be the subject of a future paper, but there are some relevant features that we can summarize as follows:

- The utilities meters are water and heat meters installed in about 7000 buildings;
- The UMDCM system collects about one million meter readings per month;
- The Representation model represents buildings, meters, meter readings and so on;
- The Trust model mirrors meters (in the most succinct form using a suitable predicate P()) and a synthetic version of meter readings;
- The properties of horizontal and vertical trust are enforced for meters and meter readings;
- Data in the Representation model are managed through a classic Transformation Rest API written in LoopBack [21] and data are stored in a MySQL database;
- The Trust model has been implemented using the Hyperledger Fabric blockchain framework [22];
- The Transaction API is a Rest API and smart contracts are written in the GO programming language [23];
- The Synchronized API has been developed using the Kong [24] API Gateway.

5. Conclusions

Blockchain is a valuable concept and a technology that will deeply change the way we foster security and reliability in IoT networks [4,25–28] and, in general, in any human-designed ecosystem by providing true dependable data. However, blockchain will not necessarily be a disruption in the way systems are designed and implemented, given the fact that trust may be smoothly "added" to existing legacy systems. It has been shown that through a simple Mirror Model, a Representation model (i.e., any ordinary current information system) may be promoted to a trusted model whose data may be conveniently used to discover patterns and make decisions. The Mirror Model is also an implicit plea to take a top-down approach toward the IoT and blockchain in general; that is, to view to the system from the point of view of Reality instead of insisting on a bottom-up approach that overvalues technicalities and leads to attitudes such as: "This technology is awesome, but to what use can we apply it?".

The concept of considering the blockchain technology from a top-down perspective is not only beneficial to the IoT domain, but it is also advantageous to all other areas where a model of reality has

to be trusted to ensure safety and security in decision-making [29]. An evident domain of application of the Mirror Model is biomedical engineering, where data from clinical devices have to be trusted in order to draw suitable decisions and conclusions in the interest of the patient. Our future research will be oriented towards the application of blockchain in cyber-physical systems including Smart Cities, the Internet of Everything (IoE) and Cyber Security and Resilience domains.

Author Contributions: The main contributions of A.B. are on Mirror Model conceptualization, the mathematical formalization, introduction, state of the art and practical consequences sections; E.B.'s main contributions are on Mirror Model conceptualizations and model revision, the state of the art and conclusions sections; M.G.'s main contributions are related on mathematical model revision and practical consequences and conclusions sections; F.P. mainly contributed on Mirror model conceptualization and on Introduction section.

Funding: This research received no external funding.

Acknowledgments: The authors would like to express their thanks to Alessandra Colombini for her assistance in reviewing and correcting this paper from the initial idea to the final draft.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ghosh, P. Machine Learning Causing "Science Crisis". *BBC News, Science & Environment*. 16 February 2019. Available online: https://www.bbc.com/news/science-environment-47267081 (accessed on 14 May 2019).
2. International Telecommunications Unit (ITU). Report on AI and IoT in Security Aspects. July 2018. Available online: https://www.itu.int/en/ITU-D/Conferences/GSR/Documents/GSR2018/documents/AISeries_Security_AspectsModule_GSR18.pdf (accessed on 14 May 2019).
3. Conoscenti, M.; Vetrò, A.; De Martin, J.C. Blockchain for the Internet of things: A systematic literature review. In Proceedings of the 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), Agadir, Morocco, 29 November–2 December 2016; IEEE: Piscataway, NJ, USA, 2016.
4. Ali, M.S.; Vecchio, M.; Pincheira, M.; Dolui, K.; Antonelli, F.; Rehmani, M.H. Applications of blockchains in the Internet of Things: A comprehensive survey. *IEEE Commun. Surv. Tutor.* **2018**. [CrossRef]
5. Panarello, A.; Tapas, N.; Merlino, G.; Longo, F.; Puliafito, A. Blockchain and IoT integration: A systematic survey. *Sensors* **2018**, *18*, 2575. [CrossRef] [PubMed]
6. Comi, A.; Fotia, L.; Rosaci, D.; Sarnè, D. A partenership approach to improve QoS on federated infrastructures. *Inf. Sci.* **2016**, *367*, 246–258. [CrossRef]
7. Banerjee, M.; Lee, J.; Choo, K.K.K. A blockchain future for internet of things security: A position paper. *Digit. Commun. Netw.* **2018**, *4*, 149–160. [CrossRef]
8. Khan, M.A.; Salah, K. IoT security review, blockchain solutions, and open challenges. *Future Gener. Comput. Syst.* **2018**, *82*, 395–411. [CrossRef]
9. Bellini, E.; Luddi, C.; Cirinnà, C.; Lunghi, M.; Felicetti, A.; Bazzanella, B.; Bouquet, P. Interoperability knowledge base for persistent identifiers interoperability framework. In Proceedings of the Eighth International Conference on Signal Image Technology and Internet Based Systems (SITIS), Naples, Italy, 25–29 November 2012; pp. 868–875.
10. Bellini, E.; Cirinnà, C.; Bergamin, G.; Messina, M.; Messuti, R. NBN:IT The Italian trusted persistent identifier infrastructure. *Int. J. Knowl. Learn.* **2014**, *9*, 347–363. [CrossRef]
11. Herrmann, M.; Petzold, J.; Bombatkar, V. Blockchain-backed analytics. Adding blockchain-based quality gates to data science projects. In Proceedings of the CARMA 2018—2nd International Conference on Advanced Research Methods, Valencia, Spain, 12–13 July 2018.
12. Bellini, E.; Ceravolo, P.; Damiani, E. Blockchain-based e-Vote-as-a-Service. In Proceedings of the IEEE International Conference on Cloud Computing (CLOUD 2019), Milan, Italy, 8–13 July 2019. accepted.
13. Yaga, D.; Mell, P.; Roby, N. *Blockchain Technology Overview*; NISTIR 8202; NIST: Gaithersburg, MD, USA, 2018.
14. Fernandez-Carames, T.M.; Fraga-Lamas, P. A review on the use of blockchain for the internet of things. *IEEE Access* **2018**, *6*, 32979–33001. [CrossRef]

15. Chowdhury, M.J.M.; Colman, A.; Kabir, M.A.; Han, J.; Sarda, P. Blockchain Versus Database: A Critical Analysis. In Proceedings of the 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), New York, NY, USA, 1–3 August 2018.
16. BigchainDB 2.0. Available online: https://www.bigchaindb.com/whitepaper/bigchaindb-whitepaper.pdf (accessed on 21 May 2019).
17. Meyer, B. *Introduction to the Theory of Programming Languages*, 1st ed.; Prentice-Hall: Bergen County, NJ, USA, 1990.
18. Landerreche, E.; Stevens, M. On Immutability of Blockchains. In Proceedings of the 1st ERCIM Blockchain Workshop 2018, Reports of the European Society for Socially Embedded Technologies, Amsterdam, The Netherlands, 8–9 May 2018.
19. Kupferman, O.; Pnueli, A. Once and for all [temporal logic]. In Proceedings of the Tenth Annual IEEE Symposium on Logic in Computer Science, San Deigo, CA, USA, 26–29 June 1995; IEEE Comp. Soc. Press: Los Alamitos, CA, USA, 1995; pp. 25–35.
20. Lu, D.; Huang, D.; Walenstein, A.; Medhi, D. A secure microservice framework for IoT. In Proceedings of the 2017 11th IEEE Symposium on Service-Oriented System Engineering (SOSE), San Francisco, CA, USA, 6–9 April 2017; Volume 1, pp. 9–18.
21. LoopBack. Available online: https://loopback.io/ (accessed on 14 May 2019).
22. Hyperledger Fabric. Available online: https://www.hyperledger.org/projects/fabric (accessed on 14 May 2019).
23. Go programming language. Available online: https://golang.org/ (accessed on 14 May 2019).
24. Kong API Gateway. Available online: https://konghq.com/kong/ (accessed on 14 May 2019).
25. Di Pietro, R.; Salleras, X.; Signorini, M.; Waisbard, E. A blockchain-Based Trust System for the Internet of Things. In Proceedings of the 23nd ACM on Symposium on Access Control Models and Technologies, SACMAT '18, Indianapolis, IN, USA, 13–15 June 2018; pp. 77–83.
26. Ouaddah, A.; Elkalam, A.A.; Ouahman, A.A. Harnessing the power of blockchain technology to solve IoT security & privacy issues. In Proceedings of the Second International Conference on Internet of Things, Data and Cloud Computing, ICC '17, Cambridge, UK, 22–23 March 2017. Article No. 7.
27. Casado-Vara, R.; de la Prieta, F.; Prieto, J.; Corchado, J.M. Blockchain framework for IoT data quality via edge computing. In Proceedings of the 1st Workshop on Blockchain-Enabled Networked Sensor Systems, BlockSys'18, Shenzhen, China, 4 November 2018; pp. 19–24.
28. Brotsis, S.; Kolokotronis, N.; Limniotis, K.; Shiaeles, S.; Kavallieros, D.; Bellini, E.; Pavue, C. Blockchain solutions for forensic evidence preservation in IoT environments. In Proceedings of the 2019 IEEE Conference on Network Softwarization (IEEE NetSoft), Paris, France, 24–28 June 2019.
29. Bellini, E.; Nesi, P.; Pantaleo, G.; Venturi, A. Functional resonance analysis method based-decision support tool for urban transport system resilience management. In Proceedings of the IEEE International Smart Cities Conference (ISC2) (ISC2 2016), Trento, Italy, 25–29 November 2016.