Blockchains through ontologies: the case study of the Ethereum ERC721 standard in OASIS*

Giampaolo Bella, Domenico Cantone, Cristiano Longo, Marianna Nicolosi Asmundo, Daniele Francesco Santamaria

Abstract Blockchains are gaining momentum due to the interest of industries and people in *decentralized applications* (Dapps), particularly in those for trading assets through digital certificates secured on blockchain, called tokens. As a consequence, providing a clear unambiguous description of any activities carried out on blockchains has become crucial, and we feel the urgency to achieve that description at least for trading. This paper reports on how to leverage the *Ontology for Agents, Systems, and Integration of Services* ("OASIS") as a general means for the semantic representation of smart contracts stored on blockchain as software agents. Special attention is paid to non-fungible tokens (NFTs), whose management through the ERC721 standard is presented as a case study.

1 Introduction

The last decade reports vast interest on blockchain technology and related applications from various realms, including economic, social, business, and academic ones. Beyond the financial speculation concerning cryptocurrencies, the interest in blockchain technologies is mainly motivated by the fact that they realize decentralized and publicly shared ledgers, where third-party intermediaries demanding the client's total and unquestioned trust are no longer required. Blockchain technologies [7] were precisely introduced to allow users to interact and run programs in a

Giampaolo Bella, Domenico Cantone, Marianna Nicolosi Asmundo, Daniele Francesco Santamaria Department of Mathematics and Computer Science, Viale Andrea Doria, 6, 95125, Catania, ITALY, e-mail: {giampaolo.bella, domenico.cantone, marianna.nicolosiasmundo, daniele.santamaria}@unict.it

Cristiano Longo, The Sicilian Wheat Bank, Via Piazza Armerina, 30, 94100, Enna, Italy, e-mail: longo@bancadelgrano.it

* This work has been supported by the ONTOCHAIN NGI European project grant agreement no. 957338. We are thankful to the ONTOCHAIN Consortium who mentored and assisted the research.

distributed way without the requirement of trusted entities, yet guaranteeing ownership, transparency, traceability, availability, continuity, and immutability of digital shared assets. Applications of blockchain technologies range from the *Internet of Things* (IoT) and robotics [18], to commerce, healthcare, insurance, energy, laws, and communications.

One of the most popular applications of Turing-complete blockchains such as Ethereum [15] is the *smart contract*. Smart contracts are self-executing and immutable programs, autonomously running and verified on a distributed and decentralized public network, which implement decentralized applications on blockchain systems called *Dapps*. In 2020, Dapps have particularly grown as an exchange tool for non-fungible tokens (NFTs), namely digital certificates stored on the blockchain representing predetermined rights on certain unique assets. NFTs are mainly used as a proof of ownership of physical or digital products. Such tokens are routinely exchanged by users to witness that assets whose uniqueness is hard to demonstrate (for example, digital images) are owned in an exclusive way. At the end of 2020, the market capitalization of NFTs reached the amount of 338 millions of U.S. dollars.[2] However, one of the main limitations of blockchains is the hard-coded nature of the transactions stored on them. As a consequence, it is hard to probe a blockchain, for example, to find smart contracts trading specific NFTs that satisfy certain requirements in terms of quality or quantity. Therefore, a formal semantic knowledge representation capturing the blockchain smart contracts as well as the activities carried out on it facilitates the understanding of blockchain concepts, the interlinking with other out-of-chain information, and also formal reasoning. Moreover, a semantic representation of blockchains enables the automatic discovery of smart contracts, the interconnection of services running on different blockchains (i.e., cross-chain integration) and the integration between on-chain and off-chain services. These features turn out to be more interesting when smart contracts are implemented as mechanisms for generating and exchanging tokens. A desirable feature of token exchange systems is a precise and intelligent query mechanism capable of determining what, when, and how certain assets have been generated, exchanged or destroyed. For example, intelligent systems may be aware of the activation of smart contracts for generating tokens with specific characteristics, e.g., of the type of exchanged asset, of the exchange of particular tokens at certain conditions, or of their destruction. More in general, intelligent systems may be aware of the activation of smart contracts and of all the related activities over the blockchain.

Beyond a semantic representation of transactions and information stored in blocks, a real, semantically represented blockchain is effectively achievable if smart contracts are interpreted as reactive agents operating on a common environment, with a fully specified semantics of available operations, committed actions, and stored data.

The representation of blockchain actors requires ontological capabilities for fully representing agents and their interactions in a detailed way. This paper adopts the *Ontology for Agents, Systems, and Integration of Services* ("OASIS") towards the full, semantic representation of the Ethereum blockchain and the smart contracts

[2] https://www.statista.com/statistics/1221742/nft-market-capitalization-worldwide/ (last access: 08/07/2021).

running on it, with a special focus on the smart contracts compliant with the ERC721 standard[3] for NFTs management.

The paper is organized as follows. Section 2 presents related work. Section 3 outlines OASIS, whereas Sections 4 and 5 depict the ontology implementing the OASIS representation of, respectively, the Ethereum blockchain and the ERC721 standard. Finally, Section 6 draws some conclusions and delineates future research directions.

2 Related works

Interest in symbiotically combining semantic web technologies and blockchains is quite recent [8, 4]. One of the areas of investigation concentrates in developing a characterization of blockchain concepts and technologies through ontologies and of blocks and transactions meta-data. An ontological albeit theoretical approach at blockchain representation exists [11]. Ruta *et al.* propose a blockchain framework for *semantic web of things* (SWoT) contexts settled as a *Service-Oriented Architecture* (SOA), where nodes can exploit smart contracts for registering, discovering, and selecting annotated services and resources [14].

Blockchain technologies are also exploited as a secure and public storage system for small data, including linked data, and to realize a more resilient architecture for the Semantic Web [8].

Other works aim at representing ontologies within a blockchain context. In [10], ontologies are used as common data format for blockchain-based applications such as the proposed provenance traceability ontology, but are limited to implementation aspects of the blockchain.

Fill discusses blockchains applied for tracking the provenance of knowledge, for establishing delegation schemes, and for ensuring the existence of patterns in formal conceptualizations using zero-knowledge proofs [9].

A semantic interpretation of smart contracts as services bases on the *Ethon* ontology [12] exist [1]. The main limitation of that approach is the poor semantic description of smart contracts, thus hindering the discovery of unknown smart contracts and of the related operations fulfilled during their life-span.

The *Blockchain Ontology with Dynamic Extensibility* (BLONDiE) project [16] provides a comprehensive vocabulary that covers the structure of different components (wallets, transactions blocks, accounts, etc.) of blockchain platforms (Bitcoin and Ethereum) and that can be easily extended to other alternative systems.

Finally, in [6] the authors illustrated how the ontology OASIS is applied for describing digital contracts (called *ontological smart contracts*), intended as agreement among agents, and how they can be secured on Ethereum.

In this contribution, the definition of digital contracts in [6] is extended to include both smart contracts over blockchains, intended as programs running on the

[3] https://ethereum.org/it/developers/docs/standards/tokens/erc-721/

blockchain and interpreted as digital agents in the OASIS fashion, including their operational semantics, and tokens exchanged through them.

3 Preliminaries

The *Ontology for Agents, Systems, and Integration of Services* ("OASIS")[4] [5] is an OWL 2 ontology for representing agents and their activities. On one hand, the ontology models (web) agents and, in particular, the way they interact and operate in a collaborative environment, regardless of the framework and language adopted for their implementation. Agents are mainly represented by means of the mentalistic notion of agent behaviour inspired by [3], encompassing goals and tasks that are achievable (either publicly available or exposed on request) by the agent, together with actions, sensors, and actuators used to perform operations. On the other hand, OASIS is used to define actions that may be requested to other agents and their related information such as operation inputs and outputs. Such requests are submitted by exchanging suitable fragments of OASIS, whereas agents whose capabilities are compatible with the requested actions are discovered by means of SPARQL queries performed over their behaviours. OASIS was applied to build a TRL3 prototype of a home assistant that activates and manages applications, devices, and users interacting with each other within the environment [5]. OASIS was also used to define agent agreements and store these in the IPFS file system [13] in order to reduce the transaction data stored directly on blockchain [6].

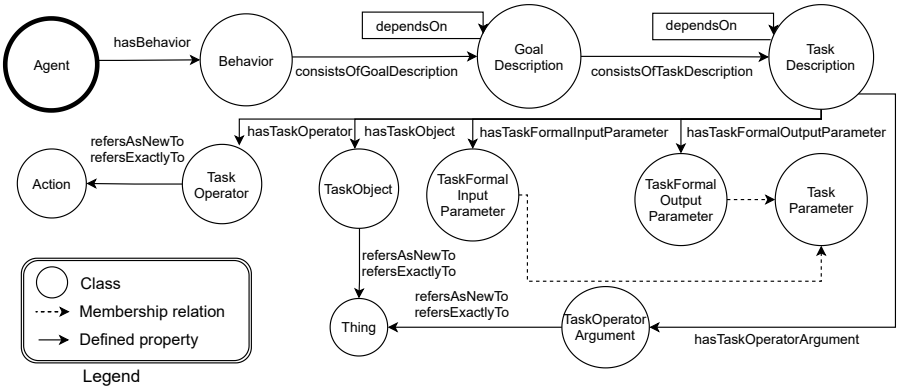

Fig. 1 Agent representation in OASIS

OASIS models agents by publicly representing their behaviours. By exposing behaviours, agents report to the communication peers the set of operations that they are able to perform and, eventually, the type of data required to execute these and the expected output. The representation of agents and their interactions in OASIS is

[4] The ontology is reachable at http://www.dmi.unict.it/oasis.owl

carried out along three main steps. The first step consists in *defining the template* for agent behaviour: templates are high-level descriptions of behaviours of abstract agents that can be implemented to define more specific and concrete behaviours of real agents. For example, a template may be designed for agents to sell and ship products to buyers, and it may be implemented by any phone seller that ships its products using the Fedex courier: templates are useful to guide developers to define the most suitable representation of their agents. The second step consists in *defining a behaviour* by implementing a template, which requires a specification of the full operational details about the sought behaviour. In the final step, agent actions are represented through ontology patterns analogous to the ones of the corresponding agent behaviours and linked to them.

As depicted in Fig. 1, agent behaviours are represented by the goals to achieve, which in turn are related with their constitutional elements, namely tasks. Tasks represent atomic operations that agents execute, and are described by the actions to be performed. Actions are drawn from a shared and common vocabulary, and can be simple or composed, eventually associated with requested input parameters and expected outputs. Finally, agent behaviours in OASIS may be associated with conditionals [6], adding constraints on the execution of actions and ensuring that certain conditions are verified before or after a task is executed. OASIS conditionals are OWL sentences that have the fashion of *Semantic Web Rule Language* (SWRL) rules [17], describing operations that must be triggered when certain conditions hold.

4 Representing Ethereum through OASIS

In this section, we describe how the Ethereum blockchain is modelled in OASIS.[5] OASIS provides a different representation of blockchains with respect to Ethon [12] and BLONDiE [16], since the description of blockchain has to be aligned with the definitions of agent and agent behaviour.

Ethereum is represented in OASIS by following the schema illustrated in Fig. 2. Ethereum blocks embedding transactions are represented by instances of the class *EthereumBlock* (subclass of *BlockchainBlock*) and connected to the transactions contained in them by means of the object-property embeds. Each Ethereum transaction is identified by an instance of the class *EthereumTransaction* (subclass of *BlockchainTransaction*), encapsulating all the transaction information.

Block miners are identified by instances of the class *EthereumNode* (subclass of *BlockchainNode*) and linked to the mined block through the object-property mines: instances of *BlockchainNode* are also instances of the class *Agent*, representing agents and provided with a behaviour as in the OASIS fashion. Moreover, instances of *BlockchainNode* (resp., *EthereumNode*) are connected to instances of the class *System* (resp., *EthereumSystem*) by means of the object-property constitutes. Such a characterization of nodes, blocks, and transactions is the main difference with

[5] The ontology is reachable at
https://www.dmi.unict.it/santamaria/projects/oasis/sources/ether-oasis.owl

analogous approaches such as Ethon and BLONDiE, since it allows one to describe activities carried out by both in-chain and out-of-chain agents, thus providing a higher-level model of the two ecosystems and a means to unify them. Specifically, Ethereum activities are classified into two main categories, namely, deployments of smart contracts, represented by instances of the class *EthereumSmartContratCreation* (subclass of *BlockchainSmartContractCreation*), and interactions with smart contracts, represented by instances of the class *EthereumSmartContractInteraction* (subclass of the *BlockchainSmartContractInteraction*).

In OASIS, smart contracts deployed into the Ethereum blockchain correspond to agents with well-defined behaviours: interactions with smart contracts are represented by OASIS *plan executions* and linked to the behaviour that induced the action. Specifically, a smart contract creation is represented by an instance of the class *BlockchainSmartContractCreation*, which is related with the description of the agent describing its behaviour by the object-property describes, the latter represented by an instance of the class *BlockchainSmartContractAgent* (subclass of the class *Agent*). Instances of *BlockchainSmartContractCreation* are also associated by means of the object-property associatedWith with the related Ethereum accounts, represented by instances of the class *EthereumSmartContractAccounts* (sublcass of the class *BlockchainSmartContractAccount*, which, in turn, is a subclass of *BlockchainAccount*). Users are instead associated with *Ethereum externally owned accounts* (EOA) represented by instances of the class *EOA-EthereumAccount* (sublcass of *EOA-BlockchainAccount*, which is a subclass of *BlockchainAccount*).

OASIS identifies four main general categories of smart contract agents: a) smart contracts providing non-fungible token exchange mechanisms compatible with the Ethereum standard ERC721, which are represented by instances of the class *EthereumERC721SmartContractAgent* (subclass of the class *NonFungibleBlockchainSmartContractAgent*); b) smart contracts providing fungible token exchange mechanisms compatible with the Ethereum standard ERC20, which are represented by instances of the class *EthereumERC20SmartContractAgent* (subclass of the class *FungibleBlockchainSmartContractAgent*); c) agents responsible for exchanging Ether cryptocurrency, which are represented by instances of the class *EtherExchangeSmartContractAgent* (subclass of the class *CryptocurrencyExchangeBlockchainSmartContractAgent*); d) general purpose and user-defined smart contract agents that do not enjoy the characteristics of the aforementioned smart contracts, which are introduced by instances of the class *GeneralPurposeBlockchainSmartContractAgent*.

In OASIS, agents may perform actions autonomously or as response to requests of executing some operations submitted by a peer. Concerning the blockchain ecosystem, we mainly limit ourselves to take into account only requests that modify the state of the chain (both internal and external), and hence induce transactions, even through *view functions*, namely functions that do not modify the state of the chain may be represented as well.

The transactions (represented by instances of the class *EthereumTransaction*) induced by interaction requests submitted to smart contracts are related with instances of the class *EthereumContractInteraction* (subclass of the class *SmartContractIn-*

teraction) by means of the object-property describes. Instances of *EthereumContractInteraction* introduce plan descriptions as in the OASIS fashion by means of the object-property describes. Plan descriptions are ways of characterizing requests of actions and the related actions performed by agents. The most notable subclass of *EthereumContractInteraction* is the class *EtherExchangeSmartContractInteraction*, representing the transferring of Ether cryptocurrency from a wallet to another. The class *EtherExchangeSmartContractInteraction* is also a subclass of the class *CryptocurrencyExchangeBlockchainSmartContractInteraction*, which in turn is a subclass of the class *BlockchainSmartContractInteraction*.

An example of representing Ethereum transactions in OASIS is illustrated in [2], Appendix A.

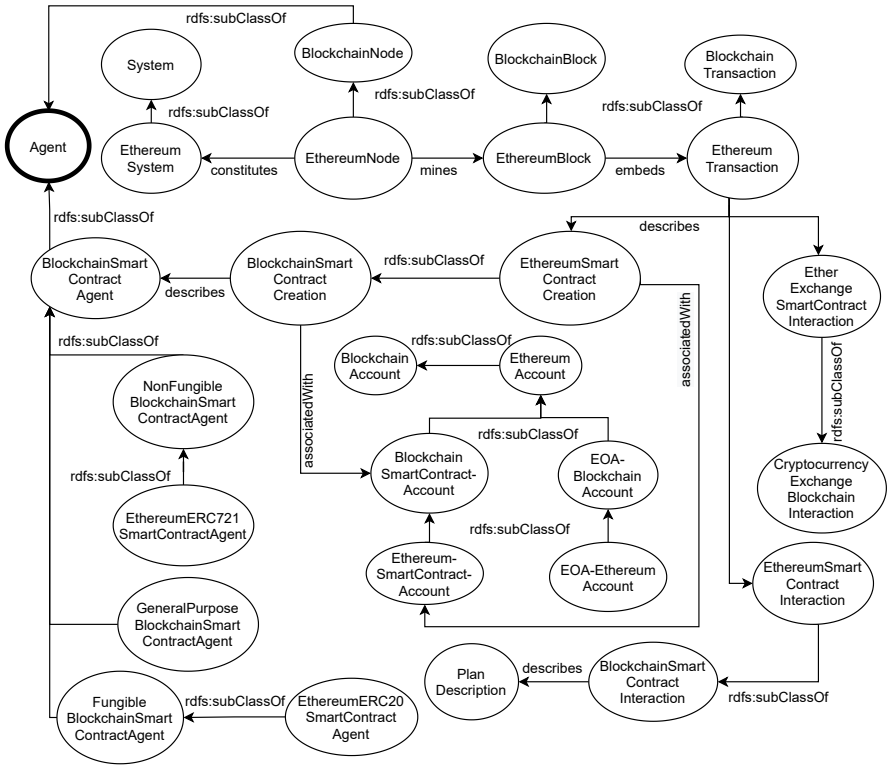

Fig. 2 Representing the Ethereum blockchain in OASIS

5 The ERC721 protocol in OASIS

In this section we show how OASIS represents the main ERC721 standard functions for managing non-fungible tokens on the Ethereum blockchain. For space limitations,

we illustrate how the ERC721 standard for minting non-fungible tokens is modelled in OASIS, whereas the ERC721 functions for transferring tokens, burning tokens, granting ownership of single tokens, of all the tokens stored in the wallet, and to verify the ownership of tokens, as represented in OASIS, are described in [2], Appendices B, C, D, E, and F, respectively.

As shown in the previous section, the smart contracts defined for the ERC721 token management are introduced by means of instances of the class *EthereumERC721SmartContractAgent* (subclass of the class *Agent*). In OASIS, agents having similar behaviours inherit the representation of their behaviour from a common template providing general descriptions that may be customized by single agents. For this purpose, OASIS provides a template for the ERC721 standard introduced by the individual *ethereum_ERC721_smart_contract_behavior_template*, which describes the behaviours of agents minting, burning, and transferring Ethereum NFTs according to the guidelines of the standard. Other functions admitted by the standard ERC721, namely the *approve* function (delegating wallets for managing single tokens), the *setApprovalForAll* function (delegating wallets for managing all the tokens owned), and *ownerOf* (for retrieving the owner of a given token), are also represented in OASIS.

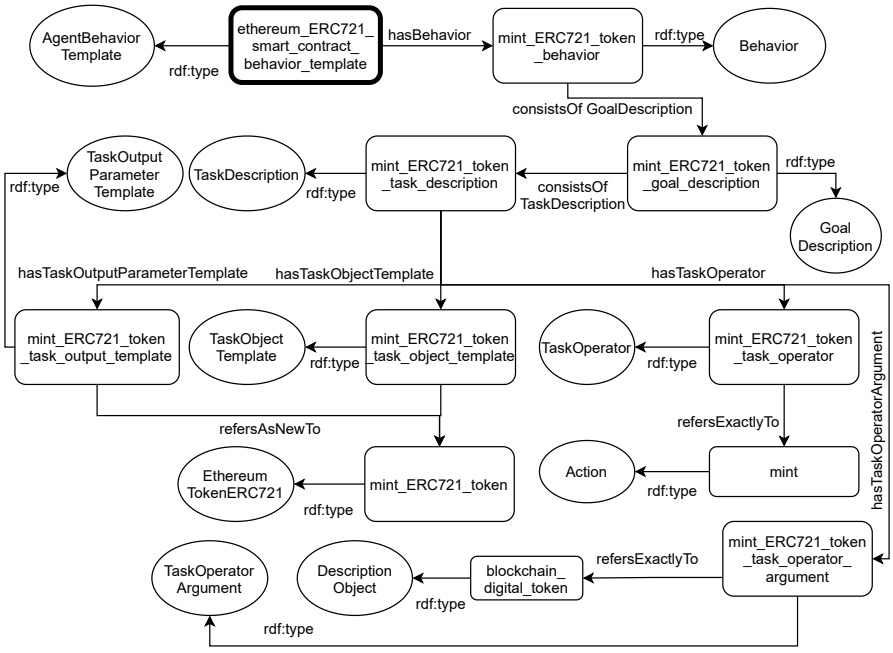

Fig. 3 OASIS behaviour template for the ERC721 minting function

Fig. 3 illustrates the behaviour template provided by OASIS describing the ERC721 function for generating new tokens. The behaviour template consists of a single goal, which in turn consists of a single task. Tasks comprise four elements. The first element is the task operator, providing the description of the action to be

performed, namely minting. The latter is introduced by means of the individual *mint* (instance of the class *Action*), through the the object-property refersExactlyTo. We recall that in OASIS the object-properties refersExacltyTo and refersAsNewTo are introduced to describe the way how constituting elements of agent behaviours must be matched when a verification of compatible behaviours occurs. The object-property refersExactlyTo introduces well-known entities whose IRIs must correspond to the IRIs of the matched entities or for which the OWL object-property sameAs has been expressed. On the contrary, the object-property refersAsNewTo introduces entities (instances of the class *ReferenceTemplate*) that are used as general descriptions encapsulating the features that the matching entities must satisfy.

The second element of the ERC721 token minting task is the operator argument introducing the individual *blockchain_digital_token* by means of the object-property refersExactlyTo. Operators and operator arguments identify unambiguously that the referred operation consists in the generation of (digital) tokens on the blockchain. The third and the fourth elements represent the recipient and the outcome of the operation, respectively. The recipient is introduced by means of a template of the task object, whereas the outcome is introduced by means of a template of the output parameter of the task. The object template and the output parameter are both connected through the object-property refersAsNewTo to the entity *mint_ERC721_token*, which describes the features that the recipient of the action must have, i.e., being an instance of the class *EthereumTokenERC721*.

6 Conclusions

This paper leveraged the OASIS ontology towards the representation of the Ethereum blockchain and the smart contracts deployed on it. Specific focus was on those that comply with the ERC721 standard for NFTs management. OASIS is exploited as a means of semantically representing Ethereum with the aim of probing the blockchain for locating smart contracts and related NFTs by specifying the desired features. In particular, the ontological representation of OASIS allows one to find smart contracts and related NFTs by inspecting their behavioural descriptions through purposely crafted SPARQL queries. It was already clear that the OASIS approach to semantic representation had the power of generality but our findings demonstrate it at an applied level.

Future work is dense. The very next step is to extend OASIS so as to model the standard protocols ERC20 and ERC1155 for fungible and semi-fungible tokens, respectively, and to represent different blockchains such as Stellar Lumens. Moreover, we intend to take up the design of a search engine exploiting OASIS to find desired smart contracts and tokens using a mechanism of auto-generating parametric *ad-hoc* SPARQL queries that could be borrowed from sibling applications [5]. The present work supports the claim that the potential of the semantic representation of blockchains has much to be unveiled in the near future.

References

1. Baqa, H., Truong, N., Crespi, N., Lee, G., Le Gall, F.: Semantic smart contracts for blockchain-based services in the internet of things. In: 2019 IEEE 18th International Symposium on Network Computing and Applications (NCA). pp. 1–5 (09 2019). https://doi.org/10.1109/NCA.2019.8935016
2. Bella, G., Cantone, D., Longo, C., Nicolosi-Asmundo, M., Santamaria, D.F.: Blockchains through ontologies: the case study of the ethereum ERC721 standard in OASIS (extended version). CoRR **abs/2109.02899** (2021)
3. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents Multi Agent Systems **8**(3), 203–236 (2004)
4. Cano-Benito, J., Cimmino, A., García-Castro, R.: Towards blockchain and semantic web. In: Abramowicz, W., Corchuelo, R. (eds.) Business Information Systems Workshops. pp. 220–231. Springer International Publishing, Cham (2019)
5. Cantone, D., Longo, C.F., Nicolosi-Asmundo, M., Santamaria, D.F., Santoro, C.: Towards an Ontology-Based Framework for a Behavior-Oriented Integration of the IoT. In: Proceedings of the 20th Workshop From Objects to Agents, 26-28 June, 2019, Parma, Italy, CEUR Workshop Proceeding Vol. 2404. pp. 119–126 (2019)
6. Cantone, D., Longo, C.F., Nicolosi-Asmundo, M., Santamaria, D.F., Santoro, C.: Ontological smart contracts in OASIS: Ontology for agents, systems, and integration of services. In: To app. in: Proceedings of IDC 2021, The 14th International Symposium on Intelligent Distributed Computing, 16-18 September, on-line (2021)
7. Christidis, K., Devetsikiotis, M.: Blockchains and Smart Contracts for the Internet of Things. IEEE Access **4**, 2292–2303 (2016)
8. English, M.D., Auer, S., Domingue, J.: Blockchain technologies & the semantic web: A framework for symbiotic development. In: Lehmann, J., Thakkar, H., Halilaj, L., Asmat, R. (eds.) Computer Science Conference for University of Bonn Students. pp. 47–61 (2016)
9. Fill, H.: Applying the concept of knowledge blockchains to ontologies. In: AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering (2019)
10. Kim, H., Laskowski, M.: Toward an ontology-driven blockchain design for supply-chain provenance. Int. Syst. in Accounting, Finance and Management **25**(1), 18–27 (2018)
11. de Kruijff, J., Weigand, H.: Understanding the blockchain using enterprise ontology. In: CAiSE (2017)
12. Pfeffer, J., Beregszazi, A., Li, S.: Ethon - an ethereum ontology (2016), available on-line: https://ethon.consensys.net/index.html
13. Protocol Labs: The Interplanetary File System (IPFS), https://ipfs.io/
14. Ruta, M., Scioscia, F., Ieva, S., Capurso, G., Pinto, A., Di Sciascio, E.: A blockchain infrastructure for the semantic web of things. In: 26th Italian Symposium on Advanced Database Systems (SEBD 2018) (2018)
15. Szabo, N.: Formalizing and securing relationships on public networks. First Monday **2**(9) (1997)
16. Ugarte Rojas, H.E.: A more pragmatic web 3.0: Linked blockchain data. In: Google Scholar (2017)
17. World Wide Web Consortium: SWRL: A Semantic Web Rule Language Combining OWL and RuleML (2004), http://www.w3.org/Submission/SWRL/
18. Xu, X., Weber, I., Staples, M., Zhu, L., Bosch, J., Bass, L., Pautasso, C., Rimba, P.: A taxonomy of blockchain-based systems for architecture design. In: Software Architecture (ICSA), 2017 IEEE International Conference on. pp. 243–252. IEEE (2017), http://design.inf.usi.ch/sites/default/files/biblio/icsa2017-blockchain.pdf