# Instrumenting Accountability in MAS with Blockchain

Fernando Gomes Papi<sup>1</sup>, Jomi Fred Hübner<sup>1</sup> and Maiquel de Brito<sup>2</sup>

 <sup>1</sup> Federal University of Santa Catarina, SC, Brazil fernando.papi@posgrad.ufsc.br, jomi.hubner@ufsc.br,
<sup>2</sup> Federal Institute of Education, Science and Technology, RS, Brazil maiquel.brito@rolante.ifrs.edu.br

**Abstract.** In this paper, we investigate a proposal where Blockchains could provide powerful tools for Multi Agent Systems. Specially regarding accountability, Blockchains could prove to be very useful tools for agents to reason upon contracts, commitments, responsibilities and so on. We list some possible approaches to *integrate* blockchains and Multi Agent Systems, by providing adequate abstractions and discussing over possible advantages and disadvantages of these different abstractions.

**Keywords:** Multi Agent Systems, Blockchain, Smart Contracts, Computational Accountability

## 1 Introduction

Multi Agent Systems (MAS) are commonly used to model complex, distributed systems such as Smart Grids, Supply Chain Management, Virtual Auctions and Marketplaces, Virtual Collaborative Enterprises, Crowdfunding Platforms, Reputation Systems and so on. These could possibly be open MAS, where there is no guarantee, by design, of cooperation among the agents in these systems. Even the opposite could be true, there could be malicious agents that freely enter and leave the MAS. In this perspective, there is the need for tools that enable accountability in the system. According to [1], accountability is "the acknowledgment and assumption of responsibility for decisions and actions that an individual, or an organization, has towards another party." The key word is responsibility: agents should be responsible for their actions, decisions and commitments. Moreover, agents are responsible for whatever agreement they put themselves through. But how can we provide agents reliable tools so that they can be accountable for their actions?

Blockchains have a good perspective on this matter because they offer agents the possibility of trustless exchange, data that is consistent and timestamped, and complete transparency and immutability. The common development tools of Multi Agent Systems could benefit from some properties provided by a blockchain. For example, there is the absolute guarantee that an interaction between agents recorded on the blockchain (as in, a message from agent A to agent B) was performed. If two agents resolve to sign a blockchain-based *Smart Contract*, there is the guarantee that this contract will be executed, even if that means penalizing an agent that breaks it. This could be a way to enforce cooperation, prevent malicious agents from harming the objective of other agents, and provide the system with logics for authorization, fairness and incentives, as described by [2].

Blockchain is currently on of the the most discussed topics in the area of financial technology. In 2009, an anonymous person (or group of people), under the pseudonym of Satoshi Nakamoto, released a white-paper [3] describing the world's first fraud-proof digital currency, the Bitcoin. The technology that enabled such technological disruption, which had been around for some time before the Bitcoin, is now known as the Blockchain, a data structure composed of chains of data blocks containing the record of all of the currency's transactions. Its cryptographic nature makes the Blockchain an unhackable time-stamped database of transactions, disseminating trust along a distributed network of nodes [4]. This characteristic enables a great number of applications that would benefit from disseminated consensus.

With this in mind, comes the questions: In terms of accountability, what could change with the integration of Blockchains and MAS? How can we successfully integrate a blockchain and MAS? What is the best abstraction from MAS for the Blockchain? If we are dealing with an environment based MAS, is the Blockchain like an ordinary environment or does it have special properties that agents should consider in their reasoning? Is it more effective if left as a generic artifact in the environment, or maybe the Blockchain would be better used if specifically tailored for the application in MAS? For example: in an auction application of MAS, the Blockchain could just receive bids and register winners, but it could also be implemented through Blockchain Smart Contracts to enforce the execution of the auction, where, in order to bid, an agent should have available funds and if it wins, the funds would be transferred automatically.

In this paper we present some initial discussion towards answering these questions and it is organized as follows. In Section 2, a quick explanation of how a blockchain works is given. In Section 3, possible approaches for the MAS and Blockchain integration is discussed considering their particularities, advantages and disadvantages. In Section 4, we present a practical example illustrating one of the proposed approaches. A brief conclusion is presented in Section 5.

Regarding nomenclature, there is an important note to be made: whenever talking about the Blockchain as a technology or system, the letter "B" will be uppercase. If we are referring to the data structure, the actual implementation of the ledger, then "blockchain" will be lowercase. Similarly, Bitcoin is written with an uppercase "B" when referring to the "system", while bitcoin with lowercase "b" refers to the tokens of the Bitcoin network, as in "the price is 2 bitcoins".

# 2 The Blockchain Technology

To the best of our knowledge, not many authors have yet proposed links between Blockchains and Multi Agent Systems, so it is worth defining what blockchains are, quickly mentioning how they work, as well as stating some terms commonly used in the blockchain community and along this paper.

In short, the blockchain is a database that has its data cryptographically signed to state its veracity. This data is packed up in blocks that will be chained by hash pointers pointing to the previous block of data. The most common data structure being used today is the Merkle Tree [5].

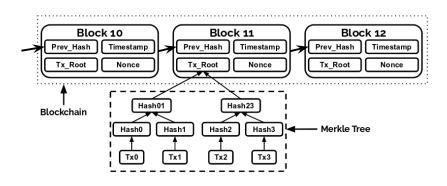


Fig. 1. Schematic of a blockchain

Figure 1 shows the structure of a blockchain. Each block contains a Merkle Tree, which is a linked tree of hashes of transactions in that block, along with a pointer that points to the previous block. The block has a timestamp and a Nonce, which is a number that was generated to approve that block by miners. Finally, it has the hash of the previous block. This means that in order to modify the data in previous blocks, it is necessary to change all the hash pointers of all the subsequent blocks to maintain consistency. In the case of Figure 1, changing data on block 10 means that the hash of blocks 11, 12... will need to be updated as well. These changes need to be replicated throughout the network, since the blockchain is essentially a peer-to-peer technology. In that case, only someone with control of at least 51% of the network will be able to do such changes. This is the reason why blockchains are considered fraud proof. The computers that execute this database are called nodes. Nodes are able to confirm if a block of transactions is valid by spending an enormous amount of computing power with the objective of solving a computational problem that can only be solved by *brute force* algorithms, generating the Nonce. This is the case of the Bitcoin ledger (some other projects offer different ways of confirming transactions). There is a reward, an incentive, for spending this computational effort (called Proof of Work, or PoW), and when a node solves the puzzle and confirms a block, this block is appended to the blockchain and propagated through the network. This process of confirming blocks is commonly called mining. There are many sources that go deeper into the mechanics of the blockchain, such as [6].

The Blockchain was already a concept by the early 90's, but only rose to prominence after the 2008 financial crisis, when Bitcoin was created. Bitcoin solved a fundamental problem with digital cash: the double spending problem [5]. Since data can be easily replicated, that is, the same token could be sent to many people, it was not trivial to develop a system where an individual can spend a given token only once. Even though some projects already tackled this problem, Bitcoin became economically viable by creating a deflationary nature. The absolute amount of bitcoins that will ever exist is 21 million, according to Satoshi Nakamoto's original proposal [3]. Therefore, there is a tendency for an increase of its face value in relation to fiduciary currencies (which are not limited in supply since abandoning the gold standard in the mid 70's) if there is an increase in demand, according to the law of supply and demand.

The Blockchain quickly evolved to be more than a safe record of transactions to be a platform for decentralized computing, in the form of Smart Contracts. The Ethereum Project [7] uses the computational power from the nodes of its blockchain for decentralized computing of functions, instead of basically wasting it. That is, one can deploy pieces of code that will be executing as long as the blockchain itself is executing in the decentralized network. Nodes will be rewarded to perform this computations. These pieces of code are the Smart Contracts. Smart Contracts are immutable, where only parameters of the operations can be changed by participants and only when there is consensus about it. If two nodes agree upon a certain contract, there is the absolute guarantee that it will execute when conditions are met. A simple example is a contract that will transfer a particular amount of tokens from participant A to participant B when a bar code is read, meaning participant A bought something from participant B. Participant A will not be able to default on the agreement with participant B, and if there are no funds in the account, then the sale is never completed in the first place. This "either wholly do some specific action, or don't do it at all" is called an atomic operation, and is a powerful characteristic of Smart Contracts.

This particular concept of Smart Contracts could be very valuable to MAS applications. Currently, the Bitcoin protocol does not support Smart Contracts. The most developed platform for this technology is the Ethereum Project, though some other platforms are being evolved to support Smart Contracts, such as Decred [8].

### 3 Integration Models

In this Section we consider possible models of integration between Blockchain and MAS. First, in Section 3.1, a generic model is proposed, where the Blockchain serves as a tool for the secure propagation of messages among agents. Then, from Sections 3.2 to 3.2.3, we explore some more complex approaches where the blockchain is part of the environment where the agents act.

#### 3.1 Blockchain as a means of communication

The first Bitcoin transaction block ever created contained the message "The Times 03/Jan/2009 Chancellor on brink of second bailout for banks" [9]. This message was the headline of the London, UK, newspaper "The Times", on January  $3^{rd}$  2009. This headline announced that the Government was on the verge of using taxpayers' money to bail bankers out of the crisis they had created. We cannot prove that the semantic content of the message is true. But we can prove, as long as a copy of the Bitcoin blockchain exists, that a transaction was made to the address "1A1zP1..." and this fact is what makes the blockchain such a promising technology. The owner of this address will never be able to state that his address did not receive 50 bitcoins upon the creation. After another 6 transactions occur, it is statistically impossible to change what was recorded on the blockchain [10]. The first transaction in the Bitcoin ledger supports the idea that blockchains can be used as effective and reliable means of communication.

Disregarding the economic value of the transactions in the blockchain, we could propose a Blockchain where each and every transaction would cost exactly 1 token (as an incentive for mining and confirming transactions), but it should contain a message that is structured in a predefined way, that is, it belongs to the formal semantics of a language. When agent A makes a transaction to agent B, agents A and B will never be able to argue that no message was ever exchanged between them. They might strongly disagree on the content of the message or the consequences this message might imply, but the existence of the triple  $\langle Message, Interlocutor, Receptor \rangle$  is undeniable. This small example supports the idea that agents can use the blockchain as a way to exchange messages in an accountable way. It would be the most basic usage of a blockchain in a MAS, supporting a reliable means of communication among agents. It is reliable because the record of message exchanges will exist as long as a sufficient number of agents hold a copy of the blockchain. Technically speaking, the blockchain will exist if at least one agent executes it. But this agent might be able to change its contents, since it is the only node in the network. If there are two nodes in the network, and one node does not accept a change made by the other, then effectively there are two different systems being executed. If there are three nodes, when two nodes accept a change in the blockchain, the third node will be compelled to accept this change as well, in order to keep its records updated and its chain valid. This is the process of achieving consensus in the

network. Figure 2 illustrates this proposal, where agents use the blockchain to securely exchange messages between them.

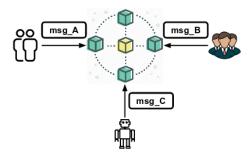


Fig. 2. Model of the Blockchain as a channel of communication between agents

However, this approach falls short in terms of accountability. Herlihy and Moise [2] bring the definition of two types of accountability: proactive and reactive. The proactive type of accountability happens upon *authorization*, such as in a scenario where agent A is authorized to make a transaction or send a message to agent B. The reactive kind would be the scenario where an agent is accountable for its actions after it has performed them. As stated by the authors, reactive accountability is of limited value if agents do not have adequate mechanisms to penalize other agents (or hold them accountable) for their actions. That is, since this approach only states that a message was exchanged between agents, only the reactive kind of accountability is achieved. Agents will still need extra tools to assess accountability. This creates the opportunity for blockchains to be more than message loggers in MAS, mainly because blockchains have evolved from distributed databases to distributed autonomous computation, in the form of Smart Contracts.

### 3.2 Blockchain as MAS environment

It is rational to make a case for a blockchain based environment for MAS when we look back at the history of mankind itself. Ian Grigg [11] states that blockchains are the third revolution in accounting. Thousands of years ago, primitive societies came up with the method of writing down transactions. Before that, humans hunted and farmed and were able only to barter these goods among a limited range of other people, normally within the same village or tribe. When the single-entry bookkeeping emerged, along with the usage of currency, numbers and writing, it was now possible to record the exchange of values and goods between a wider range of people and keep record of who owes what to whom. This created the possibility of a more complex system of commerce, which commonly boosts human societies development. The double-entry bookkeeping method, created in the medieval republics and kingdoms of modern Italy, enabled even more complexity in commerce. It allowed for better tracking of errors and audition of exchanges, since every transaction was both an *asset* and a *liability*, and both columns of assets and liabilities must sum to the same value. With a more robust method of transaction recording, goods were able to flow from one empire to the other.

As commerce and transactions grow more complex, better ways of auditing transactions are needed. Companies and governments commit fraud in their accounting books and are rarely caught or punished. Thus, the accountability of today's system is broken. Blockchain is a new revolution in accounting because it is a triple-entry bookkeeping system. Agent A transacts with agent B, and a large number of agents confirm that the transaction is valid, though none of these agents actually know who are A and B, nor A and B know who are the agents who confirmed the transaction. This anonymity in the chain of transactions adds another layer of security to the system. This is possible only through the peer-to-peer replication of the ledger, while cryptography keeps the consistency along these replications. The Blockchain is a solution to the Byzantine Generals Problem, common in distributed computing [12].

This time, rather than emulating human societies' social realities and technologies to evolve MAS, we are able to develop a blockchain model for MAS on the qo, that is, while it is also being applied and improved in different use cases. This scenario presents more opportunities than using blockchains as the channel of communication between agents, as previously proposed. Today, it is generally accepted that the environment is an essential part of the development of Multi Agent Systems, its justification being derived from many different subjects. Classic AI not only brings the concept of environment, but also defines agents as anything that perceives, through sensors, and acts upon its environment through effectors, such as the definition brought by Russel&Norvig [13]. In the MAS domain, artifacts have been proposed as first class entities to compose the MAS environment in order to achieve better coordination between agents and to provide them with information about and functionalities in regard to their environment, allowing for enhanced cognition. Artifacts are said to be reactive entities to provide functions that make agents cooperate in MAS, and shape the environment according to the system's needs [14].

In the sequence, we discuss possible places for the Blockchain in the environment shared by agents.

#### 3.2.1 Blockchain as a generic environment

Blockchains could be modeled as the whole environment of the MAS. The Ethereum Project is the first prominent provider of Smart Contracts and will be utilized in this work. All the artifacts that agents need or have access to could be coded as Smart Contracts in the chosen Blockchain platform. The advantage to this approach is largely the simplicity of developing every needed artifact on the same platform. Also, every single action or transaction made by agents will be recorded and immutable, therefore accountable. Whether an agent is delegating a task to another agent, sending another agent a particular amount of currency or simply stating a fact, everything will be forever recorded, as long as the blockchain exists.

But this does not come cheaply. Blockchain transactions are expensive regarding computational effort. Specially for large public permissionless blockchains, such as Bitcoin or Ethereum. For example, a simple Bitcoin transaction takes at least 10 minutes to be executed, and could take up to one hour to be considered fraud-proof (also said to be "confirmed", when at least 6 blocks of transactions have been linked to the blockchain after the block containing the transaction was added). Today, it operates in the speed of about 2.5 transactions per second.

Scalability is definitely Blockchain's main challenge for mass adoption today. The Bitcoin ledger was larger than 100GB by the end of 2016 and some implementations of the Ethereum's ledger were larger than 25GB by early 2017. As the technology gains adoption, this problem will only worsen. Satoshi Nakamoto's idea on this was that Moore's Law would take care of both processing power and storage needed to run a full node of the Bitcoin ledger. But maybe he did not foresee the large attention Bitcoin has drawn in less than a decade. With the concentration of mining in just a few "players" running mainly in China and Russia, the development of the system became slower and more difficult, specially when there are conflicts of interest between miners, developers and users.

This can be a huge drawback for the approach of implementing every artifact in the blockchain. The whole system could become impracticably slow, with every single action needing confirmation before being executed. This does not make sense, specially when we think of human societies: we do not register every single interaction we have with other humans or objects. We register only very special interactions, normally those that involve promises with serious consequences, contracts, transactions of valuable assets and so on. This brings the idea that perhaps a Blockchain is better modeled for MAS as one artifact available to the agents, and not the whole of the environment, in order to register special interactions that agents find suitable. Figure 3 illustrates this proposed model, where agents will only have access to artifacts and other agents through the blockchain.

#### 3.2.2 Blockchain as a single artifact in the environment

We have discussed the idea of using a blockchain as the technology to provide the whole environment for the agents, arguing that it could be unfeasible since every action would need to be registered in the blockchain. Since it is probable

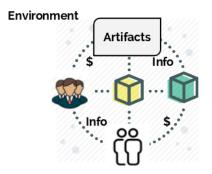


Fig. 3. Model of the Blockchain as a generic environment

that not all, but only a few, of the actions taken by agents need to be registered for accountability, we can model the blockchain as a single artifact available to the agents.

This way, agents will be able to perceive the blockchain, obtaining information from it and will be able to act by registering data on it. The abstraction provides a blockchain node running behind an artifact in the environment. Such artifact provides the agents only one type of action: blockchain transactions. By using a blockchain that does not support Smart Contracts, such as the Bitcoin protocol itself, agents are limited to transferring amounts of bitcoin and sending messages with this transaction. Though there are some limitations to sending messages in the Bitcoin protocol [15], it wouldn't be hard to conceive a blockchain that is suitable for this task. Figure 4 shows the concept of this proposal, where the blockchain is modeled as one simple transaction artifact in the environment.

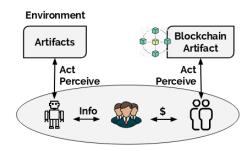


Fig. 4. Model of the Blockchain as a single artifact in the environment

This approach is fairly reasonable in cases where there is no need for extra complexity in the design of the system. In the common MAS example of an auction among agents, they can use cryptocurrencies to make payments for their bids, thus needing only to be in the sending/receiving end of a transaction. But again, this will not suffice for accountability. What if agent A, responsible for conducting the auction, does not deliver accordingly? And if agent B, winner of the auction, does not pay what was agreed upon? If it didn't have the necessary funds, why was it able to join the auction in the first place? Who will hold them accountable for the tasks they did not complete? Surely enough, all these considerations should be made upon the design of the system, in order to provide agents with the necessary tools for handling these scenarios.

### 3.2.3 Blockchain instrumenting application artifacts

Merlihy and Moir [2] cites examples of accountability problems that arise when blockchains are inserted in societies of agents, such as "because A endorsed false statement x, A can no longer be trusted with the nuclear code". These kinds of problems have long been studied by the Multi Agent Systems community, and many normative models are proposed. It is valuable noting that there is a reliable, cryptographic, fraud-proof way to solve these types of problems. Smart Contracts are a way to provide a safe logic of authorization and incentives mechanism for the agents related to the MAS.

It has been long accepted that the environment is not something that just exists, but that this environment should have a meaningful design as to support agents in their tasks [16,14]. Since Smart Contracts are programmable contracts that can be developed to perform specific functions, it makes sense to envision these Smart Contracts as specific artifacts in the environment. For example, a voting system to decide whether agent A has endorsed a false statement can be modeled to be exactly this, a voting artifact (with all the advantages of the blockchain). The nuclear code can be another artifact, with much more security embedded, for example, a strict method of access available only to agents carrying very specific roles or authorizations. These choices of design really depend on the requirements of the system and its goals. Blockchains can provide those logics of authorization, fairness and incentives.

When an agent registers itself on a contract artifact, it becomes responsible for whatever agreement this contract implies. If this contract states that all agents participating in it should vote regarding a particular issue and there is a penalization for not doing so, all agents that do not perform the required task can be automatically penalized. This means that this accountability is, in some way, automated. It is a proactive accountability. The Blockchain can create a whole new layer of trust among agents. There are many studies that propose Reputation Models for MAS. With blockchains, these Reputation Models become very powerful tools when we are dealing with Open MAS, since they cannot be tampered with. Figure 5 shows how, in this model, artifacts model the interface between agents and the blockchain. Agents are still able to flexibly communicate amongst themselves, and access other types of artifacts.

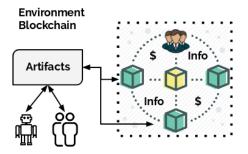


Fig. 5. Model of the Blockchain as specific artifacts in the environment

The scalability of this model depends greatly on the requirements of the system. The Ethereum Blockchain is essentially a world computer: users pay mining nodes for performing computations for them. But it is a very slow computer (somewhere around 10 to 30 executions per second). If the system requires the advantages brought by blockchains in all of its artifacts, then it must loose its requirements of speed. The number of artifacts implemented in the blockchain is not necessarily limited, but there will always be a compromise of execution time and number of interactions with the blockchain. The example of the house building auctions, presented later, makes this clear: modeling the auctions in separate artifacts makes the execution orders of magnitude slower than a simple execution of the MAS by itself (which shouldn't take longer than a couple of seconds, for example). But when actions don't need to be held accountable, they do not need to be unnecessarily incorporated in the Blockchain. For example, when the agent that creates the auctions tell other agents that auctions are open. This design trade-off will always be present.

One of the main challenges from this point on is how to design agents that will reason upon these characteristics provided by blockchains. Can we make agents distinguish the hard consequences of recording data in the blockchain rather than another database that can more easily be tampered with? This approach creates new possibilities for the advancement of Multi Agent Systems.

# 4 Case Study: Building a House

This Section presents the implementation of the model described in Section 3.2.3. We aim to demonstrate how can Smart Contracts be mapped into artifacts, and what are the necessary tools for this integration. Also, we can evaluate what is the impact of using a blockchain, a very slow system, with a much faster MAS programming platform.

For the design of the MAS, we have used JaCaMo [17], a programming language that provides abstractions of agents, environment and organizations. The Smart Contracts will be implemented in the Ethereum Platform, using the Parity [18] implementation of the Ethereum client. The communication between the MAS implemented in JaCaMo and the Smart Contracts will be done via the Web3j Java library [19]. The example is implemented in the Testing Network Kovan, for simplicity [20]. The source code is publicly available. <sup>3</sup>

The problem consists of an agent, Giacomo, that wants to build a house. Giacomo knows all the tasks that need to be done, and knows the budget for each task. Tasks are steps needed for completing the construction of the house, such as "laying the foundation", "building the floor", "building walls" and so on. Since each task has a budget, Giacomo will launch individual auctions for tasks. The whole of this example is a little bit more complex than explored in this work, involving an organization to ensure completion of tasks, but we will be focusing on the first part of the simulation, namely the *contracting phase*. Agents with the role of constructors will place bids for the tasks they want to execute. The bids will be placed directly in the blockchain, and the Smart Contract will make sure the smallest bid wins. When a given deadline is reached, Giacomo will check the Smart Contract to announce the winner for a task.

At this point, the winner will never be able to state that it wasn't in fact the winner. It has the responsibility over the commitment it made when bidding in the auction. With some complexity added, the blockchain could take care of a few extra enforcing steps, such as registering a proof of completion of the task, and the direct payment from Giacomo to the winner. The Auction artifact has the Observable Properties *task\_description*, *max\_value*, *current\_winner*, *best\_bid* and the Operation *bid*. Disagreements between agents could be resolved by looking at the data available in the Blockchain. One of the challenges resides in constructing Smart Contracts that are verifiable by all agents involved, and that any agent interacting with it can trust. For example, a Smart Contract can have its code hashed and implemented as an Observable Property. Then a copy of the same Smart Contract can be deployed in the blockchain so that agents can use it to test its functions as they wish, without further commitments. Once they are confident in the fairness of the copy, they can securely interact with the original.

As mentioned earlier, the execution time of such a system is orders of magnitude slower than a pure MAS built in JaCaMo, since each bid made by constructors need to be registered in the blockchain. For reference, this example deploys

<sup>&</sup>lt;sup>3</sup>https://github.com/FerPapi/HouseBuilding\_JaCaMo-Blockchain

10 different Smart Contracts to the network, each interfacing a different task. Each contract receives several transactions since agents are using the Operation *bid*, which requires a change in the state of the contract that must be confirmed by the network. It is also worth noting that if this was a real application deployed in Ethereum's Main Net, "real" Ethereum Tokens, called Ether, would be needed. By today's market price, each Ether costs about US\$ 300,00. It is not trivial to estimate how much this example would cost in US Dollars, but given that this is a fairly simple execution, it shouldn't be more than a few cents of the dollar. More complex Contracts that are executed for longer periods will indeed be more costly. The advantage of this cost is that a MAS integrated with a Blockchain can be used for real life applications involving transaction of assets without further complexity, such as integrating to traditional banking payment systems.

Regarding accountability, the blockchain will provide agents with reliable data and statements from the past. It can also automate the processes of payment and penalty enforcement, which makes the system more reliable overall. But blockchains will do exactly as they are programmed. So, in order to program them correctly for accountability of agents, external models and frameworks will be necessary, such as the work presented in [21]. This work also uses this house building example to demonstrate the application of an accountability protocol in MAS.

# 5 Conclusion and Further Works

Along this paper, we have discussed the adoption of a new technology into Multi Agent Systems. The blockchain is promising a revolution in accounting, finance and technology. We have briefly explained how a blockchain works and the reasons for integrating blockchain into MAS. Then we argued about approaches for modeling blockchain in a MAS: as a standalone means of communication between agents, as the environment as a whole, as a single artifact in the environment performing only transactions, or as a provider of different meaningful artifacts for the agents, going over advantages and disadvantages of each approach.

We have decided to use the blockchain to model specific artifacts in the environment, thus delegating the functionality of these artifacts to Smart Contracts. The interaction between agents and Smart Contracts is then performed through artifacts in the MAS. With this model, we implemented an integration of Ethereum's blockchain to a MAS running on JaCaMo, using a specific Java library for this purpose. The integration was successful, and an example was presented. The example was also implemented successfully, achieving its objectives. But it made clear that, when using blockchains along with MAS, there will be a trade-off between taking advantage of the blockchain's capabilities and speed. The system runs much slower when the blockchain is present, since transactions need to be confirmed by the network.

The Multi Agent Systems community has the opportunity to not only improve the MAS field itself, but also to contribute to the blockchain community with frameworks for development of pure blockchain systems. For instance, the MAS tools that implement high level abstractions as commitments [22], norms [23], and situated institutions [24] can bring significant contributions for the blockchain community. Further works in this topic would include further discussions on the most useful way of using blockchains in MAS, as well as applications in decentralized systems to simulate how agents would react and interact in the presence of this new technology. Scalability poses one of the great challenges for this new approach, and this could be tackled in further works as well.

### References

- M. Baldoni, C. Baroglio, K. M. May, R. Micalizio, and S. Tedeschi, "Computational accountability," in Proceedings of the AI\*IA Workshop on Deep Understanding and Reasoning: A Challenge for Next-generation Intelligent Agents 2016 co-located with 15th International Conference of the Italian Association for Artificial Intelligence (AIXIA 2016), Genova, Italy, November 28th, 2016., pp. 56–62, 2016.
- M. Herlihy and M. Moir, "Blockchains and the logic of accountability: Keynote address," in *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pp. 27–30, 2016.
- 3. S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.
- 4. M. Swan, Blockchain: Blueprint for a new economy. "O'Reilly Media, Inc.", 2015.
- A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press, 2016.
- A. M. Antonopoulos, Mastering Bitcoin: unlocking digital cryptocurrencies. " O'Reilly Media, Inc.", 2014.
- 7. V. Buterin et al., "Ethereum white paper," 2013.
- "Decred roadmap." available at https://medium.com/decred/2017-decredroadmap-d0da20c39db3.
- 9. "Bitcoin wiki: Genesis block." available at http://en.bitcoin.it/wiki/Genesis\_block.
- 10. "Bitcoin wiki: Confirmation." available at https://en.bitcoin.it/wiki/Confirmation.
- 11. I. Grigg, "Triple entry accounting." available at http://iang.org/papers/triple\_entry.html.
- 12. A. Miller and J. J. LaViola Jr, "Anonymous byzantine consensus from moderatelyhard puzzles: A model for bitcoin," Available on line: http://nakamotoinstitute. org/research/anonymous-byzantine-consensus, 2014.
- S. J. Russell and P. Norvig, Artificial Intelligence A Modern Approach (3. internat. ed.). Pearson Education, 2010.
- A. Omicini, A. Ricci, and M. Viroli, "Artifacts in the a&a meta-model for multiagent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 17, no. 3, pp. 432–456, 2008.

- 15. "Embedding data in the blockchain with op\_return." available at https://21.co/learn/embedding-data-blockchain-op-return/.
- D. Weyns, A. Omicini, and J. Odell, "Environment as a first class abstraction in multiagent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 14, no. 1, pp. 5–30, 2007.
- O. Boissier, R. H. Bordini, J. F. Hübner, A. Ricci, and A. Santi, "Multi-agent oriented programming with jacamo," *Sci. Comput. Program.*, vol. 78, no. 6, pp. 747– 761, 2013.
- 18. "Parity technologies." available at https://parity.io/.
- 19. C. Svensson, "Web3j." available at https://web3j.io/.
- 20. ""kovan" public testnet to provide a stable environment for ethereum development." available at https://github.com/kovantestnet/proposal/blob/master/Press%20Release.md.
- M. Baldoni, C. Baroglio, K. M. May, R. Micalizio, and S. Tedeschi, "Supporting organizational accountability inside multiagent systems," in AI\*IA 2017 Advances in Artificial Intelligence - XVIth International Conference of the Italian Association for Artificial Intelligence, Bari, Italy, November 14-17, 2017, Proceedings, pp. 403–417, 2017.
- 22. M. Baldoni, C. Baroglio, F. Capuzzimati, and R. Micalizio, "Commitment-based agent interaction in jacamo+," *Fundamenta Informaticae*, 2017.
- J. F. Hübner, O. Boissier, and R. H. Bordini, "A normative programming language for multi-agent organisations," Ann. Math. Artif. Intell., vol. 62, no. 1-2, pp. 27–53, 2011.
- M. de Brito, J. F. Hübner, and O. Boissier, "Situated artificial institutions: stability, consistency, and flexibility in the regulation of agent societies," Autonomous Agents and Multi-Agent Systems, pp. 1–33, 2017.