information

MDPI

Article

The Good, the Bad, and the Ethical Implications of Bridging Blockchain and Multi-Agent Systems

Davide Calvaresi [1,*], Jean-Paul Calbimonte [1], Alevtina Dubovitskaya [2,3], Valerio Mattioli [1], Jean-Gabriel Piguet [1] and Michael Schumacher [1]

[1] IIG, University of Applied Sciences Western Switzerland (HES-SO), Sierre 3960, Switzerland; davide.calvaresi@hevs.ch (C.D); jean-paul.calbimonte@hevs.ch (J.-P.C.); valeriomattioli580@gmail.com (V.M.); jean-gabriel.piguet@hevs.ch (J.-G.P.); michael.schumacher@hevs.ch (M.S.)
[2] HSLU, Lucerne, 6002 Switzerland; alevtina.dubovitskaya@hslu.ch
[3] Swisscom, Zurich, 8005 Switzerland
* Correspondence: davide.calvaresi@hevs.ch

Received: 31 October 2019; Accepted: 15 November 2019; Published: date

Abstract: The agent based approach is a well established methodology to model distributed intelligent systems. Multi-Agent Systems (MAS) are increasingly employed in applications dealing with safety and information critical tasks (e.g., in eHealth, financial, and energy domains). Therefore, transparency and the trustworthiness of the agents and their behaviors must be enforced. For example, employing reputation based mechanisms can promote the development of trust. Nevertheless, besides recent early stage studies, the existing methods and systems are still unable to guarantee the desired accountability and transparency adequately. In line with the recent trends, we advocate that combining blockchain technology (BCT) and MAS can achieve the distribution of the trust, removing the need for trusted third parties (TTP), potential single points of failure. This paper elaborates on the notions of trust, BCT, MAS, and their integration. Furthermore, to attain a trusted environment, this manuscript details the design and implementation of a system reconciling MAS (based on the Java Agent DEvelopment Framework (JADE)) and BTC (based on Hyperledger Fabric). In particular, the agents' interactions, computation, tracking the reputation, and possible policies for disagreement-management are implemented via smart contracts and stored on an immutable distributed ledger. The results obtained by the presented system and similar solutions are also discussed. Finally, ethical implications (i.e., opportunities and challenges) are elaborated before concluding the paper.

Keywords: blockchain; multi-agent systems; trust; reputation; ethics

1. Introduction

In the last decade, the dependency of our society on decentralized intelligent systems has dramatically escalated. Everyday items such as smartphones, wearables, vehicles, and household appliances, increasing their computation and communication capabilities, have changed habits and customs in contemporary society. In domains such as e-health, assisted living [1], telerehabilitation [2], manufacturing [3], zero-energy buildings, near-zero automotive fatalities [4], and ubiquitous computing, enforcing trusted collaborations and/or competition among (possibly heterogeneous) decentralized systems is an impending need [5].

In many cases, these decentralized systems and their interactions bear a certain resemblance to human dynamics, including the degree of autonomy in decisions and particular goals, as well as the capability to establish negotiation and collaboration among them. Therefore, some of the most employed design patterns for modeling and implementing these types of systems rely on agent oriented approaches [6]. Instances of such approaches are multi-agent systems (MAS) [7] that recently gained a crucial role in the development of decentralized intelligent systems, often exchanging sensitive data among them [8]. In such a context, accountability and trusted interactions among trustworthy agents are crucial and entail a considerable number of technical and scientific challenges [9,10]. Although a number of contributions have been made around this topic, several aspects related to security and trust in MAS are still open challenges [11–13].

Recent works [14–17] have fueled the idea of combining MAS and blockchain technologies (BCT) [18,19] in order to address some of these issues. In particular, BCT has the potential for enabling more secure, trusted, accountable, autonomous, and flexible interactions among distributed entities, removing the need for a centralized trusted third party or a trusted reputation handler. A recent systematic review [20] provided an extensive description of previous works and relevant attempts for developing models and technical implementations of BCT powered multi-agent systems. However, even if these promising results provide potential solutions for trust and security challenges in MAS, a number of undesired implications also emerge, generating new concerns that need to be addressed.

Contribution

In this paper, we study the implications of pairing MAS and BCT, especially regarding the establishment of reputation mechanisms among participating agents. In particular, we provide the design and implementation of a system based both on BCT and MAS, extending our work in [9], presenting a detailed analysis of potential ethical issues, as well as the advantages and drawbacks of combining these technologies in different scenarios. The contributions can be summarized as follows:

(i) The design and implementation of a system reconciling existing multi-agent (JADE (Java Agent DEvelopment Framework) [21]) and blockchain (Hyperledger Fabric [22]) frameworks;
(ii) The study and a development of dynamics enabling the computation of agent reputation through smart contracts;
(iii) A validation of the system in terms of agent behaviors (autonomous and user-dependent) and smart contracts (mechanisms coupled with the agent behaviors and in charge of computing and monitoring the reputation);
(iv) An analysis of the strengths, correctness, drawbacks, challenges, open opportunities, and ethical implications characterizing current and future applications connecting BCT and MAS.

The rest of the paper is organized as follows: Section 2 introduces the notions of MAS, BCT, trust, reputation, and the initial studies bridging MAS and BCT. Section 3 proposes the system's objectives, design, and components. Section 4 describes the implementation of the system. Section 5 explores the interactions and the reputation management. Section 6 analyzes the strengths, correctness, drawbacks, challenges, and opportunities in bridging MAS and BCT. Section 7 debates the ethical implications relevant to bridging MAS and BCT. Finally, Section 8 concludes the paper.

2. State-of-the-Art

2.1. Notions of Multi-Agent Systems

An agent can be rationalized as an autonomous entity observing and actuating the surrounding environment, possibly interacting with other agents [7]. Intelligent agents are also characterized by self-developed or induced objectives (driving the agent's choices to maximize its goals and performance), an amendable knowledge base, and a set of possible behaviors. Systems composed of several intelligent agents are also known as multi-agent systems (MAS), generally constituted of loosely coupled entities interconnected and organized in a collaborative/competitive network [7].

Agent behaviors are commonly related to knowledge, either private or shared with other agents via message-passing. The generation of agreements and consensus is crucial in MAS (e.g., voting systems), which often leverage on reputation, trust, and agent identity. Nevertheless, the possibility of having undesired behaviors generates minor concerns. Regardless of the distribution, dimension, and nature of the interactions (e.g., cooperative or competitive), factors such as trust and reliability are still open challenges heavily affecting the whole agent community [8,12].

2.2. Notions of Blockchain Technology

Blockchain technologies (BCT) rely on a peer-to-peer distributed ledger constituting a shared, immutable, and transparent append-only register containing all the "(inter)actions" occurring in a given network. The data (in the form of transactions) are digitally signed, broadcast by the participants, and grouped into blocks that are chronologically ordered and time stamped. Such transactions are secured by cryptographic primitives (e.g., hash function, digital signature, and encryption [23]). The content of the block is hashed, forming a unique block identifier stored in the subsequent block. The result of the hash function is deterministic and cannot be reversed. It is possible to verify if the content of the block has been modified by hashing again the content of the given block and comparing it with the identifier of the subsequent one. Every participant maintains a replica of the blockchain. Such a decentralized approach eliminates the need for a centralized trusted entity to manage the registry. BCT can execute arbitrary tasks, typically called smart contracts or chaincode logic, which are simply sets of rules written in a domain specific or a general purpose programming language [24]. A consensus protocol is required to add a new block to the ledger [25]. Based on who can access the data stored on the ledger and participate in the consensus, one can distinguish between private and public, as well as permissioned, permissionless, and hybrid BCT implementations. Depending on the chosen framework, there exist different approaches to manage the identity of the participants and their rights/duties in the consensus mechanisms.

On the one hand, in a permissionless "public" blockchain (e.g., Bitcoin [23] and Ethereum [26]), anybody can join and "write" shared states. Moreover, anybody can participate in the consensus process for determining the "validity" of a state. Permissionless blockchains are coupled with cryptocurrencies and their consensus protocols (e.g., proof-of-work (PoW)). The PoW consensus protocol was introduced in [23], in the framework of the first application of blockchain technology for Bitcoin cryptocurrency management. PoW relies on "mining", i.e., a process looking for a nonce, a random number stored in every block, so that the resulting hash of a new valid block satisfies given requirements. The difficulty threshold for finding the nonce is due to the characterization of the mining parameters. Moreover, the latter determines the average number of hashes needed to mine one block (also impacting at the energy demand level, which sometimes is unjustifiably high [27]). The security, scalability, and overall throughput of existing PoW based blockchains strongly depend on the number of transactions. For example, in order to ensure certain security guarantees of a PoW based blockchain, the maximum number of transactions per second has to be 60 [28].

In permissioned blockchains, the identities of the nodes, controlling and updating the shared state, are known. Even in the eventuality of nodes' failures, collusion, or malicious behaviors, an agreement can be reached. In the context of permissioned BCT, an overview of consensus protocols employed in different blockchain implementations (e.g., Hyperledger Fabric, Tendermint, R3 Corda, and MultiChain) was proposed by Cachin et al. [25].

Exploiting the properties of Keyless Signatures Infrastructure (KSI) is another approach to constructing an immutable ledger [29]. KSI is a distributed system to provide time stamped and server supported digital signature services. The KSI based ledger (e.g., Guardtime) employs the chain resistance property of the hash functions to verify the data integrity via the hash-chain [30]. The maintenance of the KSI based ledger is restricted to only a handful of nodes. Thus, there is no need for PoW, and the transaction throughput is high. However, the major drawbacks of such an approach are limited decentralization and trust-dependency on the maintainers of the ledger.

2.3. Notions of Trust and Reputation

Within a community characterized by agents (or entities in general) with various and heterogeneous (possibly overlapping) policies, capabilities, and roles, the interactions might depend om multiple factors such as cost functions, properties, and other values evolving over time. A reasonable assumption is to have agents with a bounded rationale. Therefore, aiming at maximizing their expected utility, agents can only rely on the information (possibly partial) they have [31]. The trust factor can minimize the effects of the uncertainty on the decision-making process.

According to Ramchurn et al. [12], interactions among agents can be characterized by (i) interaction mechanisms, (ii) who can take part in a given interaction, and (iii) when a given interaction can take place. To reduce the above mentioned risks, simplify the decision-making process, and enable autonomous interactions (in terms of "who" and "when"), it is necessary to introduce a factor/knowledge (possibly sufficient) to classify any agent's reliability.

Reliability, trust, and reputation have played a key role in MAS since the beginning (e.g., to revise conflicting beliefs and choose among opinions received from different information sources [32–35]). According to Ramchurn et al. [12], trust can be formally defined as "a belief an agent has that the other party will do what it says it will (being honest and reliable) or reciprocate (being reciprocative for the common good of both), given an opportunity to defect to get higher payoffs".

A means to establish trusted interactions is to implement reputation based mechanisms. An agent's reputation is usually computed taking into account previous behaviors, and it can be used to infer about possible future ones. Therefore, in case an agent needs to demand crucial/relevant information, the reputation can heavily impact on the decision-making process. Moreover, there is still the need to guarantee that it is properly managed (e.g., to provide an honest and verifiable way to compute it, to ensure that its historical values are tamper proof, and to make available an updated reputation value).

2.4. Combining MAS and BCT

Although recent studies showed advances on combining MAS and BCT, especially regarding theoretical contributions, there is still a need for practical, scalable, and privacy preserving system solutions [20]. According to the systematic literature research conducted by Calvaresi et al. [20], scientific contributions binding MAS and BCT started to appear in 2015. Among the main scenarios targeted by such technological synergy, we can mention collaborative governance (e.g., energy trading [14], legal accountability [36], and conflict resolution in business collaboration [16]), big data management (e.g., distributed-data anonymization [15]), coordination (e.g., swarm robotics [37], distributed AI [17], and coordination in IoT [38,39]), and trust, data integrity, and reputation management (e.g., reputation in P2P clusters [40], identity assurance [41], eCommerce [42], and supply chain [43]).

The identified drivers of combining BCT and MAS are mostly scenario dependent. However, commonly addressed aspects that span across different use-cases include accountability, transparency, and trust; for example, (i) establishing accountability, traceability, and transparency in tuple based coordination [38], (ii) coping with identification and trust [16,41,43,44], and (iii) seeking for a simplified solution for distributed master-less reputation management [40].

Even though the majority of the scientific contributions proposing to merge MAS and BCT are still at a theoretical stage, several pioneers can be acknowledged for the advancements in technical implementations. Missing a reconciling platform/framework, all the approaches leverage integrating existing technologies realizing holistic systems. Kvaternik et al. [14] proposed to conduct full and partial energy trades employing a decentralized computation fabric based on Ethereum, with on- and off-chain data management approaches. In the context of multi-level scoring systems for P2P clusters and untrusted agents, Kiyomoto et al. [40] employed BigchainDB (combining the properties of decentralized processing platforms (i.e., Ethereum) and a decentralized file system (i.e., IPFS)) [45], supporting "selfish" and self-oriented strategies. In [42], the authors modeled the interactions between supply-demand agents with ADI (anima-desire-intention) and six dimensions (physiology, belief, character, knowledge, experience, and context). Other aspects related to agreement among agents include fault tolerant consensus in MAS based blockchain systems [46].

3. Design of a MAS and BCT Based Architecture

Combining BCT and MAS requires structural and functional analysis, especially when considering the need for a platform implementation applicable in different scenarios. In order to fill this gap and to study the potential of using BCT and MAS for different types of applications, we elaborated a fully decentralized agent based design of a blockchain enabled system. The objectives of the design and implementation of such a system can be summarized as follows:

(i) identification and selection of the MAS functionalities to be integrated with or replaced by BCT;
(ii) system design and implementation, including components' integration and analysis;
(iii) test and evaluation of the system behaviors; and
(iv) critical discussion and evaluation.

From a strategic view point, in this system, we conceptually (and implementation-wise) associated the agents with the peers of a permissioned private blockchain. Such a choice was based on the assumption that the MAS required that the authenticity of data be ensured and that an agent may need to manage some sensitive information. As described further in the paper, Hyperledger Fabric (v1.0) [47] was employed as a permissioned BCT component due to its maturity (development and documentation) and its open-source availability.

Moreover, assuming that the agents can (possibly) manifest malicious behaviors, the agent's reputation was used as a discriminating factor to operate in the community (e.g., if the agent's reputation goes below a given threshold, it can be expelled from the community). For the implementation of the multi-agent environment, JADE was chosen as the MAS framework, given its compliance with the FIPA[4] standard and its simplicity in creating and handling agents [21].

In the dynamics of the agent community, JADE (MAS) and Hyperledger Fabric (BCT) are already coupled from the moment when an agent joins the network, for management of the agents' identities. To be able to interact with each other and operate on the ledger, the agents need to register and obtain the credentials. Therefore, two main classes of agents are:

BC-A: A regular agent operating in a given community, in which all the interactions are recorded on the blockchain;

CA-A: An agent handling the registration in a given agent community. In particular, the CA-A is in charge of interacting with the certification authority (CA) component of Fabric; the CA-A also offers the possibility of encoding rules and conditions for the enrollment. In the settings of multiple CAs, available in more recent versions of Hyperledger Fabric, similar multi-signature approaches can be employed to manage multiple CA-As.

Figure 1 shows the conceptual design of the system, characterizing the components and their functionalities.

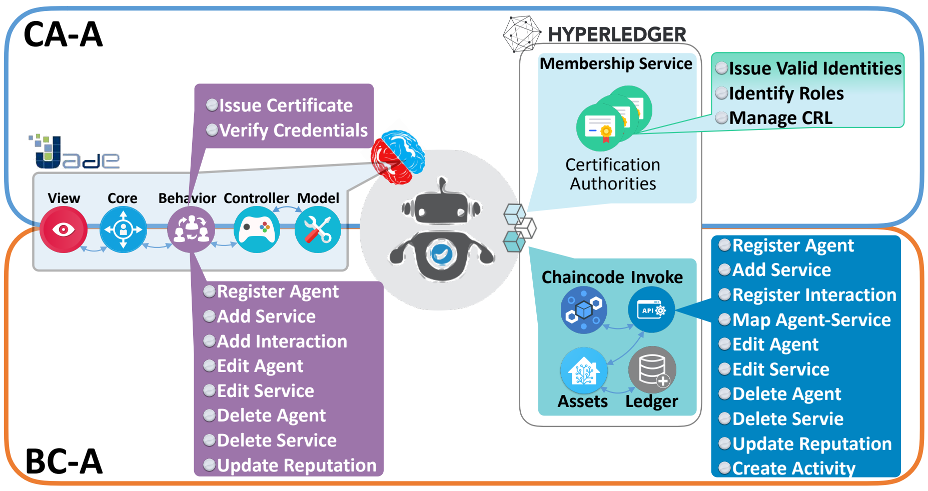


Figure 1. Conceptual design of the system components [9].

Although having different functions, both CA-A and BC-A have the same structure (see Figure 1). Each agent is composed of the following components:

- Core: the agent instance, in this case implemented in JADE.
- View: the functionality and user interface details.
- Behavior: the set of possible actions.
- Model: the adapter to operate on the underlying BCT.
- Controller: the component connecting the view and model.

The BCT components are:

- Certification authority (CA): an entity providing valid identities and certificates for the members of the blockchain network.
- Membership service: an entity that identifies CA(s) trusted to define the members of the network. An MScan identify specific roles an actor might play either within the scope of the network and sets the basis for defining access privileges [48].
- Ledger: the immutable, sequenced, tamper resistant register that records all the transactions and the database state.
- Chaincode: a program (also known as a smart contract) that is developed to interact with the ledger.
- Invoke: the APIs called when the invoke transaction is received, in this case from the agents, to process new transaction proposals.
- Assets: a collection of key-value pairs recorded as transactions in the ledger and the respective functions.

4. System Implementation

The implementation of the architecture described above constitutes a concrete example of how BCT and MAS can be implemented using currently available technologies. In particular, this implementation enhances and extends the seminal system proposed in [10]. Figure 2 schematizes the dynamics among the main components (i.e., agents, membership service, orderer, and peers running the ledger) characterizing the developed system. In this implementation, every agent ($BC - A_i$) hosts a peer that maintains the ledger and executes the chaincode. Nevertheless, it is possible to decouple agents and peers (distributing them over the network) updating the configuration files.

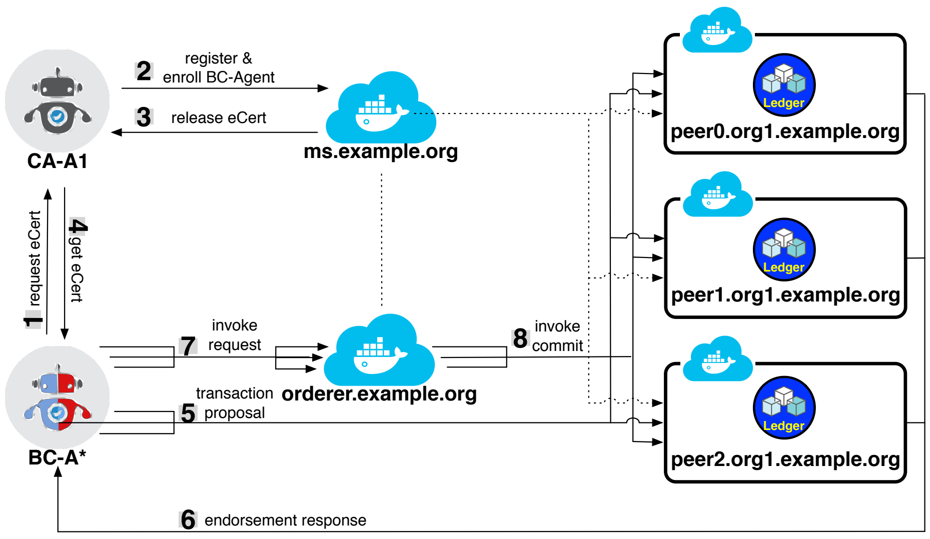


Figure 2. Design and interactions among the system components [10].

The ledger stores (i) the information about the service(s) provided by the agents (in the form of a tuple: {agent; service(s); additional info}), (ii) the information about the interactions that took place in the community, and (iii) the related evaluation from both the service's requester and executor.

It is worth recalling that in Hyperledger Fabric v1.0, the ledger consists of two distinct, though related, components: (i) the world state database (to maintain the current state of a set of ledger states) and (ii) the blockchain, the immutable sequence of blocks (containing a set of ordered transactions). At the development time, two distinct technologies were available to implement the world state database: CouchDB and LevelDB (used in the presented solution). CouchDB allows "richQueries" but cannot prevent "phantom reads". Although LevelDB does not offer the possibility to perform "richQueries" directly, it has a relational data model, supports SQL queries, and provides support for the indexing; it is not affected by "phantom reads" and can overcome the mentioned limitation by using composite keys as indices. Thus, the relational mapping was handcrafted. Figure 3 shows the representation of the system's elements in the DB.

LevelDB saves the tables in JSON format. Listing 1 shows the implementation in GO https://golang.org/ of the reputation table.

Listing 1. Reputation structure in the ledger.

```go
type Reputation struct {
 ReputationId string `json:"ReputationId"`
 AgentId   string `json:"AgentId"`
 ServiceId  string `json:"ServiceId"`
 AgentRole  string `json:"AgentRole"`
 Value   string `json:"Value"`
}
```

To allow the Level-DB world state to execute complex queries, we created a composite keys-indices mechanism. In particular, if an agent is offering a given service for the first time, the mechanism is triggered during its publication (assigning a "fair" starting value of 6/10). If an agent is demanding a given service for the first time, such a mechanism occurs after the creation of the record interaction.

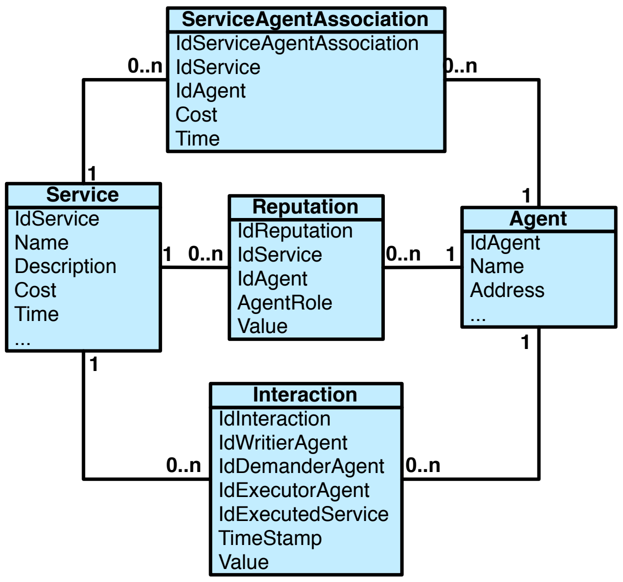


Figure 3. Relational schema of the implementation on LevelDB [9].

In the developed system, the composite key is characterized by `agentId-serviceId-agentRole-reputationId`; this key is the pointer to a given value (address: reputationId) in the DB. This enables execution of the queries such as "SELECT Agent(s) WHERE (...)" (partial-key queries). Moreover, if more complex queries are necessary, the system executes partial keys and simple queries in cascade. Such an index is not required to be associated with a value. Hence, the queried values can be extracted from the composite index itself.

When creating the composite keys-indices, the agent role (requester/executor) is checked first. Then, the previous (or initial) value of its reputation is accessed. Finally, given the evaluation of an interaction, the agent reputation is updated accordingly (see Listing 2).

Listing 2. Reputation composite keys-indices' creation.

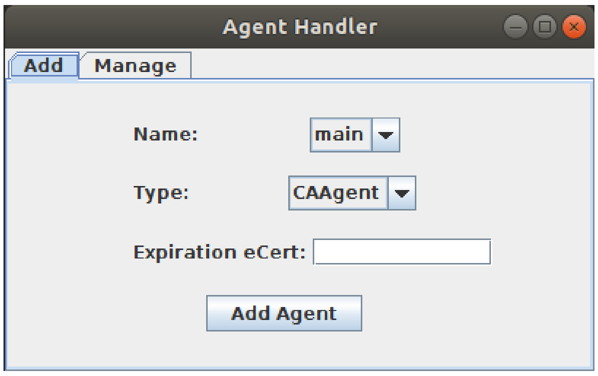
```
func CreateAgentServiceRoleIndex(reputation *Reputation, stub
shim.ChaincodeStubInterface) (agentServiceRoleIndex string, err error){
        indexName := "agent~service~agentRole~reputation"
        agentServiceRoleIndex, err = stub.CreateCompositeKey(indexName,
        []string{reputation.AgentId, reputation.ServiceId,
        reputation.AgentRole, reputation.ReputationId})
        if err != nil {return agentServiceRoleIndex, err}
 return agentServiceRoleIndex, nil}
```

In JADE, the entity connecting an agent and a service(s) offered by the agent is called the directory facilitator (DF) (http://www.fipa.org/specs/fipa00023/). This entity and the respective functions are replaced by the employment of dedicated smart contracts in Hyperledger Fabric. This design choice enables:

- avoiding a single point of failure (if the DF is unique in the community),
- reducing the response time when inquired by regular agents,
- improving accessibility and transparency,
- ensuring immutability and traceability.

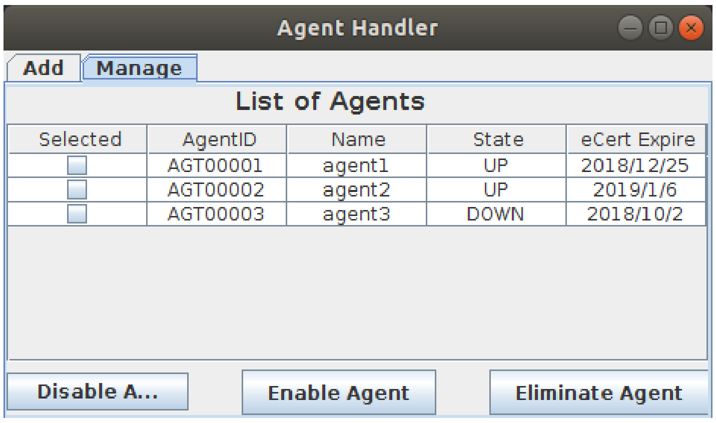
4.1. GUI

The agents' behaviors are both automated and manual (actions and choices are delegated to the user). To interact with the agents manually, and therefore with the blockchain, there is the classic command line interface and a customizable graphical interface. Figure 4 shows the interface that enables to create and register the agents (needed to test the platform). Figure 5 presents the interface to manage such agents (e.g., enabling, disabling, and eliminating agents).



Name:	main
Type:	CAAgent
Expiration eCert:	

Figure 4. View to add the agents.

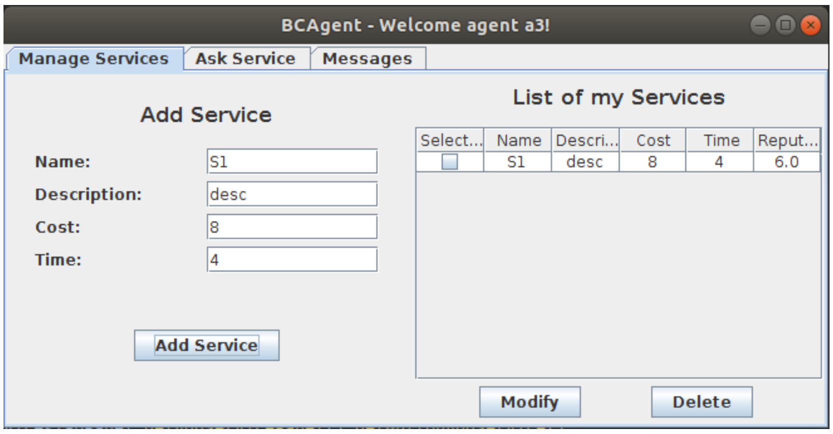


List of Agents

Selected	AgentID	Name	State	eCert Expire
☐	AGT00001	agent1	UP	2018/12/25
☐	AGT00002	agent2	UP	2019/1/6
☐	AGT00003	agent3	DOWN	2018/10/2

Figure 5. View to manage the agents.

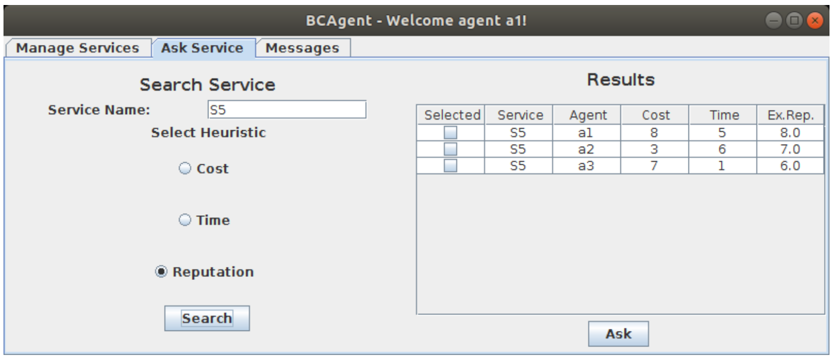
Figure 6 shows the interface that enables adding and modifying the services offered by a given agent. To require a service execution, we provide the interface depicted in Figure 7. A pre-filtering and sorting can be applied by selecting the optional features (e.g., cost, time, and reputation). Finally, an interface developed to accept or reject requests, to communicate with other agents, and negotiate the interface to compose and manage the messages is shown in Figure 8.



Name: S1
Description: desc
Cost: 8
Time: 4

Select...	Name	Descri...	Cost	Time	Reput...
☐	S1	desc	8	4	6.0

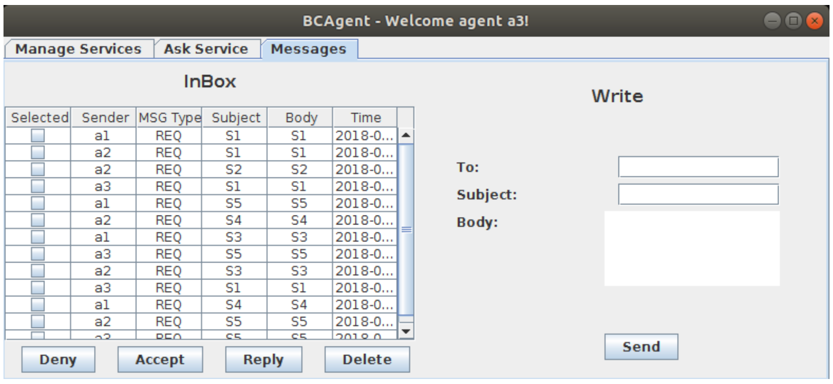
Figure 6. View to add services to the agent.



Service Name: S5
Select Heuristic
○ Cost
○ Time
● Reputation

Selected	Service	Agent	Cost	Time	Ex.Rep.
☐	S5	a1	8	5	8.0
☐	S5	a2	3	6	7.0
☐	S5	a3	7	1	6.0

Figure 7. View to search and ask for a service.



Selected	Sender	MSG Type	Subject	Body	Time
☐	a1	REQ	S1	S1	2018-0...
☐	a2	REQ	S1	S1	2018-0...
☐	a2	REQ	S2	S2	2018-0...
☐	a3	REQ	S1	S1	2018-0...
☐	a1	REQ	S5	S5	2018-0...
☐	a2	REQ	S4	S4	2018-0...
☐	a1	REQ	S3	S3	2018-0...
☐	a3	REQ	S5	S5	2018-0...
☐	a2	REQ	S3	S3	2018-0...
☐	a3	REQ	S1	S1	2018-0...
☐	a1	REQ	S4	S4	2018-0...
☐	a2	REQ	S5	S5	2018-0...

To:
Subject:
Body:

Figure 8. View to reply and send messages.

5. Interactions and Reputation Management

This section discusses how agents' behavior and reliability were interpreted using the transparent reputation management implemented in our system. First, it presents agent's possible behavior trends. Second, it discusses how they can be interpreted based on the agent's role (requester or executor of a service). Finally, it discusses the cases of missing evaluations and their potential impact on the agents' reputations.

The current mechanisms, implementing the computation and monitoring of the agents' reputation, can be extended and combined with monitoring of the service evolution over time (e.g., costs, offered performance, and availability) to enable and enhance the counter-measures in case of speculations or selfish/malicious behaviors. Therefore, we decided to associate the concept of reputation with every agent per service, as an evaluator in the role of demander and/or executor. Such a decision was also strengthened by the assumption that the agents might have different skills, thus executing diverse services with diverse qualities. Moreover, it is reasonable to assume that an agent can be able to only evaluate a given service, but not to provide it.

Thus, the implemented architecture provides:

- an overall reputation value rating the general (average) agent's reputation;
- a task specific value of a given service and role (demander/executor).

Once an agent is registered, it can require and/or provide some services. The reputation is initially set to a default value. Assuming direct interactions (e.g., CNETbased), both agents (demander and executor) must be able to evaluate the output of the interaction at its completion.

When both evaluations are provided, a smart contract is triggered, and the executor is notified with the evaluation provided by the demander. If this evaluation is significantly different from the one provided by the executor and the latter accepts to revise it accordingly, the new reputation value is computed and the ledger updated (Figure 9a). Otherwise, if the difference between the evaluation passes a given threshold, a disagreement resolution process is started (Figure 9b).

Disagreements and conflicts can happen due to the autonomy and heterogeneity of the MAS components, as well as due to possible malicious behaviors. A considerable number of scientific contributions cope with conflict resolution in MAS frameworks [49–51]. Different approaches can be employed based on the particular scenarios of using MAS, the model of expected behavior, and the outcomes of individual agents. Currently, in our system, we considered the two following scenarios:

- agreement on a specific value (with a minor variance),
- disagreement (the evaluations of demander and executor have a misalignment greater than a customizable threshold).

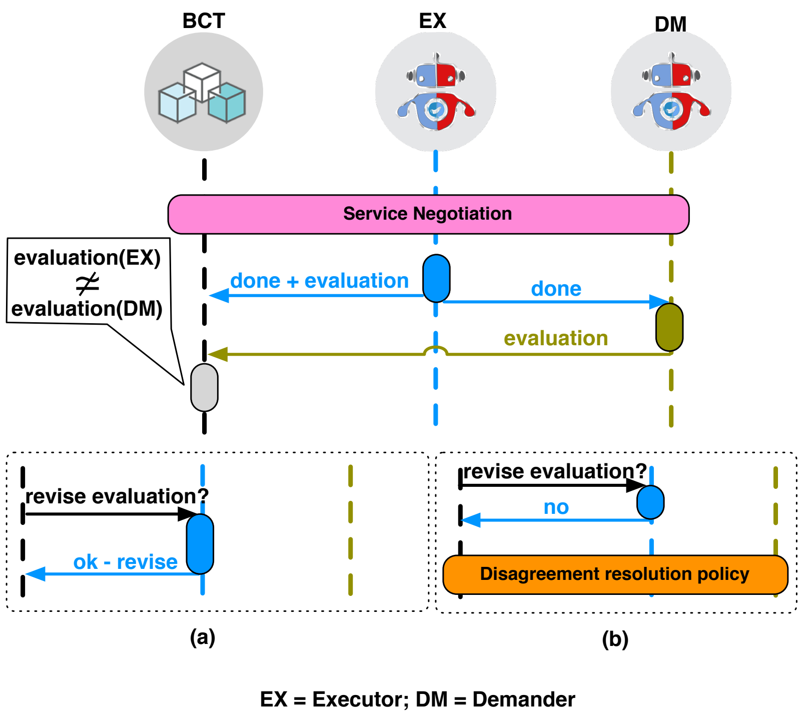


Figure 9. Evaluation management via BCT [9]. (a) The executor revised its position, thus solving the disagreement and updating the ledger, (b) Disagreement not solved. Further disagreement resolution policies activated.

In the case of disagreement, we employed the simple approach that consisted of the following steps: (i) proposing the agents to revise the evaluations (enables identifying unintentional mistakes) and (ii) consulting another agent with a higher reputation. One has to notice that it is possible to employ other existing approaches and policies for disagreement resolution.

To handle the disagreement resolution, the system provides APIs at the agent level (JADE). By doing so, it is possible to trigger several investigations upon the disagreeing values. From the smart contracts side, the function is deployed to support the investigations by checking the evolution (over time) of the reputations of both demander and executor. It enables the detection of the behavior patterns that can lead to malicious trends, systematic errors, or just to a single fault.

Figure 10 shows basic reputation trends that can be used to characterize the agent's behavior. We assumed that at the moment of the registration, every agent gets a "fair" reputation value of six out of 10. Implementing several dynamics and varying the nature of the agents' behaviors, we were able to identify three simple types of trends (see Figure 10).

Figure 10A corresponds to the case when the reputation remains constant. Such behavior can occur if (i) the agent's performance is stable or (ii) there was no activity (i.e., no service requested or provided or the agent was off-line). Figure 10B depicts the case when the reputation values are increasing and approaching the maximum reputation value threshold (10 in our simplified example). In Figure 10C, the reputation of the agent decreases, approaching the lower bound threshold. The latter, combined with the inclination of the curve, can be used when making the decision of excluding the agent from the community.

While Figure 10 shows only the behavioral trends, the actual reputation may be a composition of these trends, as presented above. Figure 11 shows a possible scenario, where each point of the graph corresponds to the punctual reputation value based on the evaluations provided by both demander and executor concerning a given interaction. Evaluation methods and the quality of the performance offered by the agents can present considerable variability, which is reflected in the fluctuations of the agents' reputations.

The behavioral trends can be interpreted based on the agent's role. Table 1 summarizes the possible interpretations for the executor (Exe) and demander (Dem) for every trend identified and discussed above. For instance, if the reputation value of the executor grows over time, this agent can be seen as a reliable service provider, constantly improving the quality of its service. However, in the case of the composition of the trends, the interpretation may not be straightforward. Yet, outliers perturbing the reputation's trend can be used to detect malicious behavior, a single mistake of the service provider, or unintentionally made errors when providing evaluations.

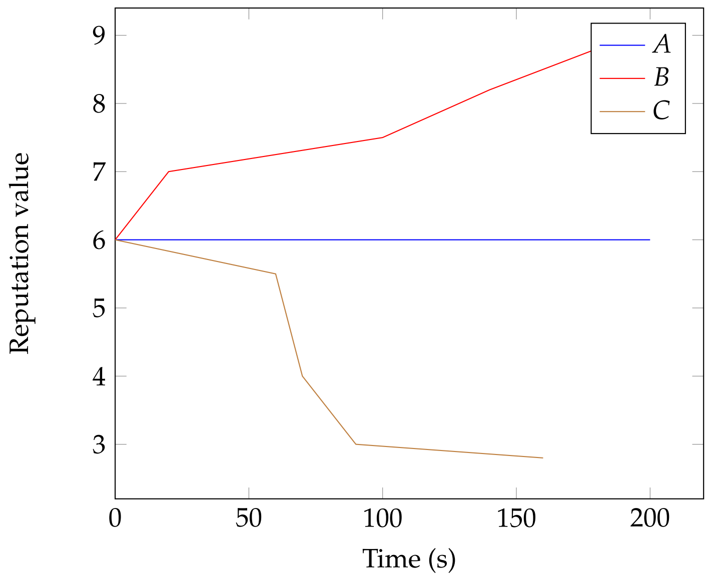


Figure 10. Basic reputation trends.

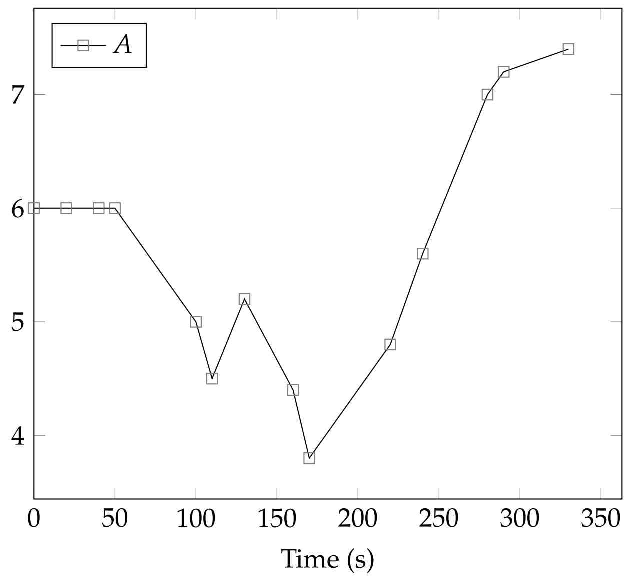


Figure 11. Composite reputation trends.

Table 1. Possible interpretations of the identified reputation trends. Exe, executor; Dem, demander.

	Figure 10A	Figure 10B	Figure 10C	Figure 11A
Exe	Stable	Bad evaluators(s), Performance in decline	Good Executor	Unpredictable, Time limited problem
Dem	Stable	Bad executor(s) Possible malicious intentions	Good evaluator	Honest mistake, Single bad action

The relevance of the trend also relies on the time (for how long a given value is maintained) and on the number of evaluations (number of interactions and evaluations occurring). The combination of the slope of a curve with the number of points also plays an important role. These data can be used to detect agent's intentions, unintentional mistakes in evaluation, as well as the inactivity of an agent.

Currently, we are analyzing potential corner cases that can occur when our system is in use. These include a missing evaluation from a demander (currently, there is no mechanism to enforce the demander to provide an evaluation), an executor (it can be off-line, and therefore, the evaluation will never be provided), or both. We are also investigating the possibility to put timing constraints for providing evaluations, in particular, for the executor. Introduced in a smart contract, such constraints can enable identification of the situations when the deadline of the execution of the service is missed and the need to update the reputation accordingly.

6. Discussion

Combining existing technologies might unveil their hidden potentials or the need for the development of new reconciling technologies enabling a real brake-through in new (or not-yet-considered) application domains. Nevertheless, many questions arise, especially concerning how and when both functional and non-functional requirements can be met. The emergence of BCT and its integration in/with MAS demands clear comprehension and formalization of the impact and implications.

Summarizing the evidence from the current state-of-the-art and the presented solution, Section 6.1 elaborates on the strengths, Section 6.2 discusses the correctness and meaningfulness of binding MAS and BCT, Section 6.3 argues about the drawbacks, and finally, Section 6.4 presents the still open challenges and opportunities.

6.1. Strengths of Binding BCT and MAS

Business collaborations demanding for high reliability, shared, trusted, privacy preserving, immutable data repositories, and smart contracts' execution are already benefiting from the simple employment of BCT [52]. However, to broaden the spectrum of advantages and possibilities, coupling BCT based systems with MAS paves the way to brand new possibilities, e.g., by simplifying the distributed governance of groups of people. It may also help enhance the dynamicity of the interactions, henceforth lowering the transaction costs, for reaching an agreement, still complying with standardized rules [53]). MAS are characterized by entities driven to maximize their own cost function. Thus, malicious or unwanted behaviors may arise. Tracking the interactions of intelligent agents on an immutable registry can prevent situations where two or more parties make conflicting claims about specific transactions (e.g., whether a payment or a service has been performed). Hence, in the developed system, the disagreement resolution policy allows spotting and disambiguating misalignment concerning a service execution, while keeping immutable track of the time of that occurrence (possibly being malicious or an honest mistake).

6.2. Correctness in Binding BCT and MAS

Considering that the need for binding BCT and MAS can be questionable, also the correctness of the existing/proposed solutions has to be evaluated. For example, using BCT as a service enabling the data owner to track how his/her anonymized data are shared (e.g., in [44]) can be questionable. From one hand, while by law (EC Data Protection Directive 95/46/EC; Health Insurance Portability and Accountability Act), once the data are properly anonymized (excluding any possibility of de-identification), the data do not belong anymore to the "data owner", henceforth, BCT can introduce unnecessary burden for the system. On the other hand, blockchain technology can be used in order to track the data that have been released in anonymized form, including the anonymity level of the data (but not necessarily keeping all the data, only some metadata). This may be employed as an input to compute the risk of being re-identified if more (anonymized) data about the data owner are released. One can argue that keeping track of all the transactions can put the anonymization at risk. However, recording of all the transactions can be used to compute the risk and prevent the violation of the data owner's privacy [54].

Taking into account dependency on the application domain, assessing or justifying the correctness of a given implementation is challenging, in particular given that many prototypes (like the one presented in this paper) are still not fully tailored on/committed to a given practical application. Nevertheless, even at a proof of concept or a demonstrator stage, some considerations can be made. For example, claims such as follows can be misleading: "one of the benefits of our approach is that no central server managed by a trusted third party (TTP) is required; therefore, the cost of deployment of such system can be reduced as there is no need to maintain the TTP" [44]. Kiyomoto et al. [44] used Hyperledger Fabric (implementation of the private blockchain technology), which requires membership service and certification authority for registering users and distributing the credentials (public/private keys). While both entities can be decentralized to avoid a single point of trust (and therefore, a single point of failure), deployment of such decentralized versions can require even more efforts and costs than the maintenance of a centralized TTP.

6.3. Drawbacks of Binding BCT and MAS

On the one hand, some limitations (e.g., lack of feasibility) can be directly connected to a given technology. For instance, this was the case when Kvaternik et al. [14] developed their proposed solution relying on Solidity (https://solidity.readthedocs.io/en/develop/, a language for creating smart contracts on the Ethereum platform), which did not provide floating-point data types yet. On the other hand, some limitations follow from the new (un)desired capabilities granted by the solutions reconciling MAS and BCT (e.g., having immutable ledgers implies storing (machine readable and agent executable) contracts and obligations, which, to this extent, increase the overall complexity of the system). Moreover, concerning both the solution presented in this paper and already existing ones, a framework supporting migration from a smart towards a self-aware contract (contracts aware of their internal/external contextual state and progress and able to reason about their behavior while being a law artifact [36]) is still missing. Finally, although merging MAS and BCT is promising and strategic, some existing strengths typical of a given technology might disappear. For example, in the presented solution, while features such as identity management, persistence, and tamper-proof mechanisms gained great benefit, the scalability property (typical of MAS) is hampered by the constraints imposed by the available implementation of Hyperledger Fabric.

6.4. Challenges and Opportunities in Binding BCT and MAS

The design and implementation proposed in this paper make a step forward towards the synergy between MAS and BCT. Nevertheless, employing existing technological frameworks/architectures, there is a lack of instruments required for seamless integration of the two philosophies, e.g., in the case of using public permissionless BCT implementations, there is a need to improve the nodes'/agents' anonymity. In both collaborative and competitive behaviors, trust and accountability are crucial. The system proposed in this study relies on reputation management to foster transparency and traceability (henceforth trust). Even if the results regarding feasibility and performance are promising, it is still challenging to develop commercial software, so as to satisfy real-world requirements. Hence, although plugging BCT "as-is" into a MAS can help to cope with privacy and identity issues, it is not yet a sustainable and definitive solution.

Permissioned BCT can be used to enforce role based access control and enable the management of privacy policy. Using off-chain cryptographic primitives, secure multi-party computations, and anonymous communications (e.g., onion routing) can further support anonymity and privacy requirements. However, particularly pronounced in the case of permissioned BCT, the lack of scalability weakens their adoption in MAS. Therefore, in the envisioned future, reconciling technology, addressing such an issue is of high priority.

When user anonymity is desired, elimination of the risks of de-identification of the users is challenging (due to the possibility of tracking and linking the transactions to a user). Therefore, non-discriminative and privacy preserving MAS based and BCT equipped systems for trading are still missing. Thus, developing new economic models and the formalization of threat and security models are required.

This work, aligned with the current state-of-the-art, underlines the following open challenges: (i) to create legal foundations for BCT in distributed intelligent domains; (ii) to enable the verification (and correctness) of the chaincode/smart contracts; (iii) to prevent colluding and malicious behaviors (e.g., creation of mining pools); (iv) to develop mechanisms ensuring privacy and anonymity when necessary; (xv) to facilitate the development and adoption of the new BCT; (vi) to merge the management of certification authorities and membership service (among MAS and BCT); (vii) to address the scalability issues of BCT; (viii) to define which types and data have to be on-/off-chain (depending on the scenario); and (ix) to ensure the reliability of the mechanisms binding MAS and BCT.

Merging MAS and BCT represents a great potential, opening a broad set of new challenges. However, employing BCT is not always necessary and justified. Hence, the indiscriminate use of BCT may only over-complicate the system [55]. Once the need for BCT is ensured, coping with the early stage related issues becomes mandatory. Norta et al. [36] recently proposed a language for self-aware contracts (SAC). It is a static and declarative approach relying on obligations. Nevertheless, relating declarative and imperative programming in smart contracts is still an open challenge.

7. Ethical Implications of Binding MAS and BCT in Real-World Scenarios

The binding of BCT with MAS in real-world scenarios opens precise ethical opportunities and related challenges. The potential contradictions between BCT and GDPR requirements and between

BCT and environmental sustainability exceed the scope of this article. Therefore, this section focuses on the ethical implications of the empowerment promised by the combination of BCT and MAS.

At first glance, the primary societal promise of the system presented in this paper (embedding BCT within an MAS) offers a new perspective of individual empowerment. Indeed, the improvement in transparency and security brought by BCT makes trusted third parties (TTP) unnecessary in two ways: (i) TTP are not required any longer to ensure the historical value of the data of the transactions, and (ii) given the smart contract technologies, the need for TTP is also excluded from the agreements' stipulation among the users. Such a condition significantly extends the contractual freedom of single users, whose, at the moment, interactions with unavoidable TTP (e.g., platforms such as Airbnb and Uber) are far from being egalitarian. For example, in the context of legal disputes, open access to a secured history of the transactions turns out to be crucial for individuals, particularly relevant in MAS, where negotiations are numerous, fast, and complex. Thanks to the use of BCT in MAS, the informational and executive asymmetry that underpins the business model of many platforms comes here to an end.

BCT eradicates the monopoly on the information and ensures a certain degree of equality in control of the contracts' implementation. Nevertheless, such features offered by BCT are conditions "necessary" but not "satisfactory" to fully meet the collective empowerment of users. Indeed, two limits to the above empowerment strategy can be mentioned: (i) How should a user know that another would be ready to pass a contract with the need for a reminder (e.g., a virtual assistant)? An intermediary might still be needed to match supply and demand. Thus, the end of informational asymmetries and monopolies cannot be expected only from BCT. Possible inequalities can be overcome once users (people or agents) have achieved an agreement. (ii) Although smart contract based technologies prevent malicious and erroneous agreements' implementations, they do not eliminate the risks of mistakes. Indeed, users are demanding that the decisions involving them or made on their behalf (as a contracting party) have to be explainable. Henceforth, transparency without explainability is no longer acceptable [56,57]. Summarizing, the benefits of binding BCT and MAS in terms of empowerment are both significant and uncertain.

The first challenge related to the "empowerment promise" of BCTs in MAS is to implement them in a way that the (secure and available) information could be effectively used by a single individual (person or agent) who is not supported by an army of lawyers and data scientists.

The second challenge is associated with the hope of removing the need for intermediaries, especially centralized ones. In particular, through the use of smart contracts, the liability system between contracting parties can be adapted accordingly. Such a challenge is not solely related to BCT, but it also extends to those MAS relying on smart contract technologies, where part of the implementation of a contract is delegated to agents or AI algorithms in general.

The possible risks connected with such delegation are: (i) bad implementation of a contract through AI; and (ii) besides no glaring mistake seeming to have occurred, it could be the case that the terms of a contract leave room for interpretation and do require a new consensus procedure between the contracting parties. In both cases, there will be no platform liable for possible mistakes. Moreover, the concerns about considering the two contractual parties liable for a delegated decision arise. Hence, intermediaries will be needed not only to detect mistakes (which would bring us back to the need for explainability), but also to provide a framework for a dialog between the contracting parties, if a problem occurs. These scenarios demand that MAS using smart contract technologies must have internal procedures to evaluate and review a defective AI.

Access to the distributed ledger has another critical ethical impact on the construction of mutual trust between MAS users (either virtual or humans). As mentioned above, the proposed MAS relies on the ability of the users to establish mutual trust (achieved via secure reputation building mechanisms). However, building trust through reputation is not an exclusively strategic issue. Erroneously or purposefully damaging (or hiding from other contracting parties) the agent or individual reputation is both an ethical concern and a cost/benefit problem for rational investors. From an ethical perspective, asserting if BCT actually strengthens the ability of users to trust each other in a rationalized way requires further observations. Let us first recall what is required to enable reliable reputation mechanisms. The rationalization of reputation demands for (i) the authenticity of its origin (a user X knows that a given appreciation of Y has actually been expressed by Z in given circumstances). It supposes (ii) a more general traceability (X has the proper means to trace both the origin of a declaration of Z about Y and the corresponding information on transactions between Y and Z) and (iii) accuracy (X must have good reasons to believe that Y says something correct about Z). Certainly, BCT supports the satisfaction of (i) and (ii). The public storage of information ensures traceability, while the authenticity of an assessment is better warranted by the possibility of identifying the contributor (when using a non-anonymous permissioned BCT, as in the case of the proposed system). Above all, public storage of information enables each user to see if and about what exactly two people agree or disagree. In addition, it allows building a two level assessment in an MAS: the evaluation of a given transaction and a more general assessment of someone's reputation, resulting in the previous local evaluations provided by his/her peers. However, this can raise some privacy concerns: Is it acceptable that all peer-to-peer interactions are visible for all the participants of the network?

Concerning (iii), does the reputation management advocated in this paper strengthen, in the absence of guarantees, the overall accuracy of the assessments? In case of a disagreement, or before contracting, the entities can check the (overall or specific) reputation of the others on the permissioned blockchain. The particular reputation of someone for a given service is itself evaluated through his/her more general reputation. A good general reputation of an entity is supposed to certify his/her reliability. The epistemological principle that underlies the model is that unfair behaviors will be detected over time and lead to the exclusion of the non-objective member from the group. This premise is far from being naive. On the one hand, Abramova et al. [58] proposed a study based on Airbnb. It showed that the impact of a negative review depends on the kind of critique and on the reaction of the host: "the trust-damaging impact of negative reviews is confirmed only when the object of the criticism is manageable by the host; moreover, the impact of the confession/apology strategy is nearly twice as effective as that of denial, both compared to the case when a complaint is left without any response." [59]. Summarizing, people read the comments and try to distinguish between fair and unfair criticisms. On the other hand, Mayzlin et al. [60] and Zervas et al. [61] proved that fraud is not a big matter in peer-to-peer platforms.

Nevertheless, the "collective wisdom" premise has to be balanced against other recent studies in social psychology on online rating. In particular, it has been shown that, as stated by Origgi and Pais, "where the cover of anonymity is removed, users are often reluctant to give negative evaluations, in particular when they are mildly disappointed. There are various reasons for this: they prefer to avoid conducts [62]; they fear retaliation [63,64]; they are worried about being sued for libel; or they feel bad for the seller (in particular after a personal off-line interaction [62]. Besides the under-reporting of negative review, literature shows a certain number of biased reporting [61]: herding behavior, whereby prior ratings impacts on the evaluations of subsequent reviewers; self-selection, when consumer who are a priori more likely to be satisfied with a product or a service are also more likely to buy it and then review it [65]; strategic manipulation of reviewing, for instance in that users may artificially enhance the trustworthiness of others when writing reviews because they may be friends [66]". Hence, if someone is unfortunate to have a bad rating at the beginning of his/her online career, the management of reputation, as proposed in the developed system, might create unjust harms. Clearly, although this matter is not explicitly linked to BCT and MAS, it is unavoidable. Nevertheless, BCT sets things in stone, raising the responsibility of the system' designers. For example, no matter at which level (either MAS, BCT, or both), the designers have to implement the "right to be forgotten".

8. Conclusions

The system described in this paper addressed the challenge of enforcing trust in MAS by implementing mechanisms for transparent reputation management via BCT. In particular, the paper presented the design and development of a system based on JADE and Hyperledger Fabric v1.0. Our system enforced a trustworthy community by extending the management of the agent identity to the membership service of Hyperledger Fabric, by distributing the association agent offered service(s) using the ledger, and by implementing the mechanisms to store immutably and compute transparently agents' reputations based on the evaluations of the interactions between the agents. Use cases with different agents' behaviors were used to test the proposed system. The results showed promising directions to undertake. The system appeared robust and reasonably scalable (limited only by the available Hyperledger Fabric implementation). Moreover, its graphical interface simplified the interactions, making the testing much faster and easier with respect to a classic command-line interface and providing the possibility to employ this system in various use-case scenarios. Extending the interface will enable us to obtain extensive evaluation results for specific use-case scenarios and interactions.

The application of relatively new technology, such as BCT, is challenging. The latter is not fully framed by standards and has not been widely adopted yet. Although the benefits of combining BCT and MAS are justified and a prototype of the proposed solution was implemented, some technical limitations persist. For example, elaborating on the findings of the state-of-the-art and the proposed solution, challenges, such as (i) avoiding a single point(s) of failure of the system by defining a reasonable mapping between real entities and distributed components of the underlying blockchain technology (Hyperledger Fabric), (ii) leveraging cryptographic solutions for providing better security and privacy, (iii) verifying the correctness of the implementation of the smart contract, and (iv) adopting and adapting the blockchain and agent technology in real-world systems, still need to be addressed.

Finally, we discussed the ethical considerations when combining BCT and MAS in real-world scenarios. Depending on a specific application area, the immutability property of the ledger and transparency of reputation management can empower the users of the system and support trustworthy interactions among them. Yet, at the same time, the users' privacy can be compromised, and the framework will fully rely on the smart contract, its logic and implementation, together with possible (un)intentional malfunction, pawned at the stages of design and implementation of the contracts. Striving to attain a non-discriminative environment by employing smart contracts, the BCT enabled MAS would allow one to avoid the intermediaries in conflict resolution, yet underlying the importance of addressing open questions of software reliability and verification, especially in the BCT environment.

Author Contributions: D.C. alvaresi took care of the system design, has been the main contributor to the paper, and coordinated the activities. J.P.C. supported the writing of the paper and took care of the state of the art. A.D. supported the implementation and the elaboration of the state of the art. V.M. collaborated designing the system and took care of its entire developement. J.G.P. elaborated on the ethical implications of the proposed system. M.S. supervised the activities and supported the writing of the paper.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflicts of interest

References

1. Calvaresi, D.; Cesarini, D.; Sernani, P.; Marinoni, M.; Dragoni, A.; Sturm, A. Exploring the ambient assisted living domain: A systematic review. *J. Ambient Intell. Humaniz. Comput.* **2017**, *8*, 239–257.
2. Calvaresi, D.; Marinoni, M.; Dragoni, A.F.; Hilfiker, R.; Schumacher, M. Real-time multi-agent systems for telerehabilitation scenarios. *Artif. Intell. Med.* **2019**, *96*, 217–231.
3. Hsieh, F.S. Modeling and control of holonic manufacturing systems based on extended contract net protocol. In Proceedings of the 2002 American Control Conference, Anchorage, AK, USA, 8–10 May 2002; Volume 6, pp. 5037–5042.
4. Rajkumar, R.R.; Lee, I.; Sha, L.; Stankovic, J. Cyber-physical systems: The next computing revolution. In Proceedings of the 47th Design Automation Conference, Anaheim, CA, USA, 13–18 June 2010.

5. Schatten, M.; Ševa, J.; Tomičić, I. A roadmap for scalable agent organizations in the internet of everything. *J. Syst. Softw.* **2016**, *115*, 31–41.
6. Shoham, Y. Agent-oriented programming. *Artif. Intell.* **1993**, *60*, 51–92.
7. Russell, S.J.; Norving, P. *Artificial Intelligence: A Modern Approach*; Prentice Hall: Malaysia, 2002; pp. 111–114, .
8. Calvaresi, D.; Marinoni, M.; Sturm, A.; Schumacher, M.; Buttazzo, G. The Challenge of Real-Time Multi-Agent Systems for Enabling IoT and CPS. In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, Leipzig, Germany, 23–26 August 2017, doi:10.1145/3106426.3106518.
9. Calvaresi, D.; Mattioli, V.; Dubovitskaya, A.; Dragoni, A.F.; Schumacher, M. Reputation Management in Multi-Agent Systems Using Permissioned Blockchain Technology. In Proceedings of the 2018 IEEE/WIC/ACM International Conference onWeb Intelligence (WI), Santiago, Chile, 3–6 December 2018; pp. 719–725.
10. Calvaresi, D.; Dubovitskaya, A.; Retaggi, D.; Dragoni, A.F.; Schumacher, M. Trusted registration, negotiation, and service evaluation in multi-agent systems throughout the blockchain technology. In Proceedings of the 2018 IEEE/WIC/ACM International Conference onWeb Intelligence (WI), Santiago, Chile, 3–6 December 2018; pp. 56–63.
11. Yu, B.; Singh, M.P. An evidential model of distributed reputation management. In Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1, Bologna, Italy, 15–19 July 2002; pp. 294–301.
12. Ramchurn, S.D.; Huynh, D.; Jennings, N.R. Trust in multi-agent systems. *Knowl. Eng. Rev.* **2004**, *19*, 1–25, doi:10.1017/S0269888904000116.
13. Hedin, Y.; Moradian, E. Security in Multi-Agent Systems. *Procedia Comput. Sci.* **2015**, *60*, 1604–1612.
14. Kvaternik, K.; Laszka, A.; Walker, M.; Schmidt, D.; Sturm, M.; Lehofer, M.; Dubey, A. Privacy-Preserving Platform for Transactive Energy Systems. *arXiv* **2017**, arXiv:1709.09597.
15. Qayumi, K. Multi-agent Based Intelligence Generation from Very Large Datasets. In Proceedings of the 2015 IEEE International Conference onCloud Engineering (IC2E), Tempe, AZ, USA, 9–13 March 2015; pp. 502–504.
16. Norta, A.; Othman, A.B.; Taveter, K. *Conflict-Resolution Lifecycles for Governed Decentralized Autonomous Organization Collaboration*; Proceedings of the 2015 2nd International Conference on Electronic Governance and Open Society: Challenges in Eurasia. ACM, **2015**.
17. Ponomarev, S.; Voronkov, A. Multi-agent systems and decentralized artificial superintelligence. *arXiv* **2017**, arXiv:1702.08529.
18. Swan, M. *Blockchain: Blueprint for a New Economy*; O'Reilly Media, Inc: sebastopol, CA, 2015.
19. Tapscott, D.; Tapscott, A. *Blockchain Revolution: How the Technology behind Bitcoin is Changing Money, Business, and the World*; Penguin: 2016.
20. Calvaresi, D.; Dubovitskaya, A.; Calbimonte, J.P.; Taveter, K.; Schumacher, M. Multi-agent systems and blockchain: Results from a systematic literature review. In Proceedings of the International Conference on Practical Applications of Agents and Multi-Agent Systems, Toledo, Spain, 20–22 June 2018; pp. 110–126.
21. Bellifemine, F.L.; Caire, G.; Greenwood, D. *Developing Multi-Agent Systems with JADE*; John Wiley & Sons: chichester, UK, 2007; Volume 7.
22. Androulaki, E.; Barger, A.; Bortnikov, V.; Cachin, C.; Christidis, K.; De Caro, A.; Enyeart, D.; Ferris, C.; Laventman, G.; Manevich, Y.; et al. Hyperledger fabric: A distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference, Porto, Portugal, 23–26 April 2018; p. 30.
23. Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*; 2008.
24. Pass, R.; Shi, E. Hybrid consensus: Efficient consensus in the permissionless model. In Proceedings of the LIPIcs-Leibniz International Proceedings in Informatics, Budapest, Hungary, 29 May–1 June 2017; Volume 91.
25. Cachin, C.; Vukolić, M. Blockchains Consensus Protocols in the Wild. *arXiv* **2017**, arXiv:1707.01873.
26. Buterin, V. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform. Available online: https://github.com/ethereum/wiki/wiki/%5BEnglish%5D-White-Paper (accessed on 18 November 2019).
27. O'Dwyer, K.J.; Malone, D. *Bitcoin Mining and Its Energy Footprint*; ISSC 2014/CIICT 2014; IET: Limerick, Ireland, 2013; pp. 280–285.

28. Gervais, A.; Karame, G.O.; Wüst, K.; Glykantzis, V.; Ritzdorf, H.; Capkun, S. On the security and performance of proof of work blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, 24–28 October 2016; pp. 3–16.
29. Buldas, A.; Kroonmaa, A.; Laanoja, R. Keyless signatures' infrastructure: How to build global distributed hash-trees. In Proceedings of the Nordic Conference on Secure IT Systems, Ilulissat, Greenland, 18–21 October 2013; pp. 313–320.
30. Buldas, A.; Saarepera, M. On Provably Secure Time-Stamping Schemes. In Proceedings of the ASIACRYPT, Jeju Island, Korea, 5–9 December 2004; Volume 3329, pp. 500–514.
31. Savage, L.J. T*he Foundations of Statistics*; Dover INC: New york, USA, 1972.
32. Dragoni, A.; Mascaretti, F.; Puliti, P. A generalized approach to consistency based belief revision. *Lect. Notes Comput. Sci.* **1995**, *992*, 231–236.
33. Dragoni, A.; Giorgini, P. Distributed Belief Revision. *Auton. Agents Multi Agent Syst.* **2003**, *6*, 115–143.
34. Dragoni, A.; Animali, S. Maximal consistency, theory of evidence, and bayesian conditioning in the investigative domain. *Cybern. Syst.* **2003**, *34*, 419–465.
35. Dragoni, A.; Giorgini, P. Learning agents' reliability through bayesian conditioning: A simulation experiment. *Lect. Notes Comput. Sci.* **1997**, *1221*, 151–167.
36. Norta, A.; Vedeshin, A.; Rand, H.; Tobies, S.; Rull, A.; Poola, M.; Rull, T. Self-Aware Agent-Supported Contract Management on Blockchains for Legal Accountability; *URL: http://whitepaper. agrello. org/Agrello_Self-Aware_Whitepaper.* **2017**, 89
37. Ferrer, E.C. The blockchain: A new framework for robotic swarm systems. *arXiv* **2016**, arXiv:1608.00695.
38. Mariani, S.; Omicini, A.; Ciatto, G. N*ovel Opportunities for Tuple based Coordination: XPath, the Blockchain, & Stream Processing*; 18th Workshop "From Objects to Agents". Vol. 1867. Sun SITE Central Europe, RWTH Aachen University, 2017.
39. Skowroński, R. The open blockchain-aided multi-agent symbiotic cyber–physical systems. *Future Gener. Comput. Syst.* **2019**, *94*, 430–443.
40. Gattermayer, J.; Tvrdik, P. Blockchain based multi-level scoring system for P2P clusters. In Proceedings of the 2017 46th International Conference on Parallel Processing Workshops ICPPW, Bristol, UK, 14–17 August 2017; pp. 301–308.
41. Leiding, B.; Norta, A. *Mapping Requirements Specifications into a Formalized Blockchain-Enabled Authentication Protocol for Secured Personal Identity Assurance*; International Conference on Future Data and Security Engineering. Springer, Cham, 2017..
42. Shen, J.; Shen, J.; Huang, Y.; Huang, Y.; Chai, Y.; Chai, Y. A cyber-anima based model of material conscious information network. *Int. J. Crowd Sci.* **2017**, doi:10.1108/IJCS-01-2017-0001.
43. Bonino, D.; Vergori, P. Agent Marketplaces and Deep Learning in Enterprises: The COMPOSITION Project. In Proceedings of the Annual Computer Software and Applications Conference, Turin, Italy, 4–8 July 2017; pp. 749–754.
44. Kiyomoto, S.; Rahman, M.S.; Basu, A. On blockchain based anonymized dataset distribution platform. In Proceedings of the 2017 IEEE 15th International Conference on Software Engineering Research, Management and Applications (SERA), London, UK, 7–9 June 2017; pp. 85–92.
45. McConaghy, T.; Marques, R.; Müller, A.; De Jonghe, D.; McConaghy, T.; McMullen, G.; Henderson, R.; Bellemare, S.; Granzotto, A. *BigchainDB: A Scalable Blockchain Database*; BigChainDB: Berlin, Germany, 2016.
46. Feng, L.; Zhang, H.; Chen, Y.; Lou, L. Scalable Dynamic Multi-Agent Practical Byzantine Fault-Tolerant Consensus in Permissioned Blockchain. *Appl. Sci.* **2018**, *8*, 1919.
47. Cachin, C. Architecture of the hyperledger blockchain fabric. *Workshop Distrib. Cryptocurr. Consens. Ledgers* **2016**, *310*, 4.
48. IBM. Hyperledger-Fabric Online Documentation. Available online: https://hyperledger-fabric.readthedocs.io/en/release-1.2/ (accessed on 10 September 2018).
49. Tessier, C.; Chaudron, L.; Müller, H.J. *Conflicting Agents: Conflict Management in Multi-Agent Systems*; Springer Science & Business Media: New york, USA, 2006; Volume 1.
50. Resmerita, S.; Heymann, M. Conflict resolution in multi-agent systems. In Proceedings of the IEEE Conference on Decision and Control, Maui, HI, USA, 9–12 December 2003; Volume 3, pp. 2537–2542.
51. Vasconcelos, W.W.; Kollingbaum, M.J.; Norman, T.J. Normative conflict resolution in multi-agent systems. *Auton. Agents Multi Agent Syst.* **2009**, *19*, 124–152.

52. Hull, R.; Batra, V.S.; Chen, Y.M.; Deutsch, A.; Heath III, F.F.T.; Vianu, V. Towards a shared ledger business collaboration language based on data-aware processes. In Proceedings of the International Conference on Service-Oriented Computing, Banff, AB, Canada, 10–13 October 2016; pp. 18–36.
53. Shermin, V. Disrupting governance with blockchains and smart contracts. *Strateg. Chang.* **2017**, *26*, 499–509.
54. Dumas, M.; Hull, R.; Mendling, J.; Weber, I. *Blockchain Technology for Collaborative Information Systems (Dagstuhl Seminar 18332)* Available online: http://drops.dagstuhl.de/opus/volltexte/2019/10236/ (accessed on 18 November 2019)
55. Wüst, K.; Gervais, A. *Do you need a Blockchain?* Available online: https://ieeexplore.ieee.org/abstract/document/8525392 (accessed on 18 November 2019)
56. Anjomshoae, S.; Najjar, A.; Calvaresi, D.; Främling, K. Explainable agents and robots: Results from a systematic literature review. In Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, Montreal, QC, Canada, 13–17 May 2019; pp. 1078–1088.
57. Calvaresi, D.; Mualla, Y.; Najjar, A.; Galland, S.; Schumacher, M. *Explainable Multi-Agent Systems through Blockchain Technology*; Springer, Montreal, 2019.
58. Abramova, O.; Shavanova, T.; Fuhrer, A.; Krasnova, H.; Buxmann, P. Understanding the Sharing Economy: The Role of Response to Negative Reviews in the Peer-to-peer Accommodation Sharing Network **2015**, *2*, 175–193.
59. Origgi, G.; Pais, I. Digital reputation in the mutual admiration society. *Studi Di Sociol.* **2018**, *2*, 175–193.
60. Mayzlin, D.; Dover, Y.; Chevalier, J. Promotional reviews: An empirical investigation of online review manipulation. *Am. Econ. Rev.* **2014**, *104*, 2421–2455.
61. Luca, M.; Zervas, G. Fake it till you make it: Reputation, competition, and Yelp review fraud. *Manag. Sci.* **2016**, *62*, 3412–3427.
62. Fradkin, A.; Grewal, E.; Holtz, D.; Pearson, M. Bias and reciprocity in online reviews: Evidence from field experiments on airbnb. In Proceedings of the Sixteenth ACM Conference on Economics and Computation, Portland, OR, USA, 15–19 June 2015; pp. 641–641.
63. Adamic, L.A.; Lauterbach, D.; Teng, C.Y.; Ackerman, M. Rating friends without making enemies. In Proceedings of the Fifth International AAAI Conference onWeblogs and Social Media, Catalonia, Spain, 17–21 July 2011.
64. Overgoor, J.; Wulczyn, E.; Potts, C. Trust propagation with mixed-effects models. In Proceedings of the Sixth International AAAI Conference onWeblogs and Social Media, Dublin, Ireland, 4–7 June 2012.
65. Tussyadiah, I.P. Strategic self-presentation in the sharing economy: Implications for host branding. In *Information and Communication Technologies in Tourism 2016*; Springer: 2016; pp. 695–708.
66. Lauterbach, D.; Truong, H.; Shah, T.; Adamic, L. Surfing a web of trust: Reputation and reciprocity on couchsurfing. In Proceedings of the 2009 International Conference on Computational Science and Engineering, Vancouver, BC, Canada, 29–31 August 2009; Volume 4, pp. 346–353.