Process Modeling in Decentralized Organizations Utilizing Blockchain Consensus

Felix Härer*,[a]

[a] Digitalization and Information Systems Group, University of Fribourg, Switzerland

Abstract. *Blockchain technologies allow for multiple organizations, individuals, and software to become part of a decentralized network where they can reach consensus on the state of the system including all data stored. Due to the ability of reaching consensus even among untrusted actors, the idea of decentralized organizations has been proposed, where both the system components as well as their coordination are distributed. Given such a system of distributed actors, the problem of decentralized coordination for following common goals and planning becomes apparent. This paper addresses the decentralized coordination problem by means of a blockchain-based approach that uses conceptual modeling to reach consensus in decentralized organizations. With a unified view on the processes and instances of distributed actors, the aim is a decentralized planning and execution through models. For this purpose, an existing approach for decentralized process modeling and instance tracking is applied and extended with the possibility for actors to form consensus on an organizational level through blockchain transactions.*

Keywords. Decentralization • Process Modeling • Consensus • Smart Contract • Blockchain

Communicated by Peter Fettke. Received 2019-08-29. Accepted after 1 revision on 2020-06-29.

1 Introduction

The organizational structure of the firm has evolved over the last decades to become more decentralized and open to collaboration (Piller et al. 2017; Simon et al. 1954). On a global scale, organizations today are part of inter-national and inter-connected networks, offering services in collaboration and forming supply chains. Assuming the continuation of the decentralization trend, methods and technical infrastructure for the organizational planning and execution are required to support decentralized structures to a greater degree. The proposition of this paper is the adoption of blockchain technology for this purpose, applying its ability to form consensus such that a shared representation of organizational planning and execution can be reached among decentralized actors. This representation in the form of conceptual models of the decentralized structure, business processes, and execution states can be shared and understood across organizations, and edited collaboratively by participants connected over a blockchain.

One of the promises of decentralized blockchain technologies is the formation of peer-to-peer networks, where no participant is in control of data and information alone (Buterin 2013; Nakamoto 2008; Wood 2019). Consensus algorithms ensure the emergence of a global state, which is consistent and independently verifiable by any of the distributed participants. The trusted global state acts as a dependable source of information not only to individual participants, but to the network. Due to this property, the network and its peers may be the basis of non-centrally governed organizations. Decentralized organizations based on blockchains (Hsieh et al. 2018; Scott et al. 2017; Swan 2015) are designed and operated by distributed actors and software in the form of smart contracts, acting

* Corresponding author.
E-mail. felix.haerer@unifr.ch

as peers of a network such that no single entity is in control. However, this notion of decentralized organizations has a purely technical focus that does not extend beyond the blockchain. When considering only software and smart contracts, the sociotechnical and organizational aspects are not addressed.

In general, the information system of decentralized organizations can be characterized as a decentralized system. In such a system, system components as well as their coordination are distributed (Härer 2019, p. 115 f.). The coordination of system components in the form of individual participants or software allows the system to follow common goals and plans, realized in concrete tasks and activities performed at run-time.

For designing a system of decentralized organizations, their tasks and activities carried out in processes need to be described on the basis of their structure. In particular, the following research questions (RQ) need to be answered:

RQ 1: What representations can be used for modeling the structure of distributed system components, their planned behavior at design-time, and their actions carried out at run-time?

RQ 2: How can consensus on the representations be established among the systems' components for realizing distributed coordination?

RQ 3: How can the distributed tasks and activities performed at run-time be observed in order to validate the behavior planned at design-time?

So far, concepts for decentralized governance and organizations based on blockchain consensus have mostly been discussed in light of decentralized autonomous organizations (DAO) (Dupont 2017; Mehar et al. 2019). In existing realizations, the decision-making processes are implemented through voting mechanisms of smart contracts. Smart contracts here represent autonomous, impartial and verifiable state machines for decision-making and the execution of business transactions. In its current form, the decentralized organization has a technical focus centered around decision-making and the execution of individual smart contract functions. However, blockchain consensus may be utilized also on a sociotechnical level, for linking activities across participants and software to form process-oriented systems in accordance to a particular domain. While the complexity of decentralized systems can be relatively high, as the number of distributed components is theoretically unbounded, there exist approaches for handling the resulting complexity by the design of process-oriented systems through conceptual modeling (Karagiannis et al. 2016; Sinz 2019). Here, the specification of the system is expressed in terms of semantic concepts of a given domain. One particular domain is process modeling, where the development of process-oriented systems is facilitated by specialized modeling methods.

In this paper, the research questions on the design of process-oriented systems with distributed components and non-centralized coordination are addressed by extending an existing approach. The approach (Härer 2018, 2019) describes a method for the decentralized modeling and instance tracking of processes. It is applied and extended here for reaching consensus in decentralized organizations. Integrity-secured and non-repudiable transactions conducted by the participants of a network specify sub-systems and activities over time in process models. In this way, the coordination of possible interactions given by a process definition is distributed. Decentralized blockchains that support such an architecture possess the properties of a decentralized system, i. e. they consist of distributed system components in the form of peers and a distributed coordination maintaining the blockchain by mining and consensus. Due to the immutability property and the trust-less characteristics of decentralized blockchains (Xu et al. 2017), transactions for the collaborative modeling (Liu et al. 2009) gain a binding character such that the resulting models can be regarded as contracts between the decentralized participants of the collaborative process.

The remainder of this paper is structured as follows. In Sect. 2, foundations for process modeling

(Sect. 2.1), blockchain-based modeling (Sect. 2.2), and the notion of decentralization in relation to consensus (Sect. 2.3) are motivated and discussed. Sect. 3 introduces decentralized process modeling and details the consensus-based modeling and collaboration implementation. Sect. 4 discusses characteristics of the approach and draws conclusions. Sect. 5 summarizes the results in light of decentralization with an outlook on open research questions.

2 Foundations

2.1 Model-based Design of Processes and Workflows

From an organizational point of view, decentralization describes the opposite of a concentration of organizational structures (Bleicher 1991; Christie et al. 2003; Simon et al. 1954). The participants of a decentralized organization are distributed and can act autonomously to a certain degree, however, at the same time, they are required to cooperate in order to plan and execute processes. Therefore, the foundations concern process models and their collaborative characteristics at first. Because of the involvement of participants that are distributed, the modeling of collaborative processes is related as well as the collaborative construction of models.

2.1.1 Modeling of Collaborative Processes

The design of process-oriented systems is well established as part of approaches for modeling enterprises and enterprise architectures. Modeling languages such as Business Process Model and Notation (BPMN) (Object Management Group 2014) introduce the concepts and syntax necessary for modeling processes and collaborations. Modeling methods such as the Semantic Object Model (SOM) (Ferstl and Sinz 2006) introduce means to design processes from a sociotechnical perspective. Using conceptual modeling, semantic concepts of the process domain are expressed in terms of appropriate syntax, e. g. for organizational units, task activities, and control flow. Modeling approaches facilitate the specification of choreographies and processes to a detailed level of individual tasks, possibly executed automatically by a workflow engine. Regarding the design of process-oriented systems, SOM assumes the construction of systems that possess the characteristics of distributed systems (Ferstl and Sinz 2006). The structure is initially modeled, followed by procedural views that are subsequently introduced. BPMN consists of procedural notations that model behavioral aspects of processes with the possibility of deriving a detailed workflow definition. For collaborations, BPMN contains collaboration diagrams suited for the processes of multiple organizations and their interactions (Object Management Group 2014). Through this representation, collaborative processes can be described (Fdhila et al. 2015).

2.1.2 Collaborative Construction of Models

Software tools for conceptual modeling and metamodeling are mostly rooted in a centralized paradigm. Regarding the possibilities for collaboration in existing tools, a distinction can be drawn with regard to their architecture and their communication.

- Architectures commonly employed for collaborative modeling are client-server architectures, such that modeling among multiple parties can occur interactively (Nicolaescu et al. 2018). Here, modeling might be carried out on the web, or through a database accessed by multiple users remotely (Fill and Karagiannis 2013). Peer-to-peer architectures, where any client may also act as a server, are not in focus today.
- Communication in collaborative modeling either operates in a synchronous or an asynchronous fashion. While tools such as SyncMeta (Nicolaescu et al. 2018) operate in a synchronous mode, allowing for interactive and ‘near real time’ modeling, there also exist model versioning approaches (Altmanninger et al. 2009; Brosch et al. 2012; Kelly and Tolvanen 2018) which provide a versioning-enabled repository. These systems commonly operate

asynchronously by providing explicit commit and check-out operations on the basis of version control systems such as Git.[1]

2.2 Blockchain-based Modeling Approaches

Distributed characteristics are addressed by modeling approaches that additionally take blockchains into account, defining the state of the art. Blockchain-based modeling approaches exist for business process management (Mendling et al. 2018) and primarily concern execution at this time. Specialized choreography models (Ladleif et al. 2019), process and workflow models (Evermann and Kim 2019; López-Pintado et al. 2019; Tran et al. 2018; Weber et al. 2016) or decision logic (DMN) (Haarmann et al. 2018) are implemented by smart contracts in order for them to be executed on the Ethereum blockchain. While traditional workflow engines may allow for distributed execution, e. g. (Camunda 2018), blockchain-based choreographies and interorganizational workflows can connect organizations by a global representation defined through models. A distinction can be made between approaches which use given smart contracts as an execution engine (Ladleif et al. 2019) and generative approaches, which generate smart contract code in a model-driven fashion (Di Ciccio et al. 2019; López-Pintado et al. 2019; Tran et al. 2018; Weber et al. 2016).

The permissioned management of enterprise models in a blockchain has been suggested (Fill 2019; Fill and Härer 2018) in order to track changes and to prove the ownership, provenance and existence of models or model elements. Here, the blockchain is operated by a group of appointed miners while anyone is able to validate it.

On the Ethereum blockchain, the attestation of conceptual models has been suggested recently (Härer and Fill 2019b). Similar to decentralized identity applications, models do not need to be held available on the chain in cases where an assertion on their existence can be made instead. In this way, relatively low cost and size of individual transactions can be achieved.

Not in focus of existing approaches is the decentralized decision-making and planning of processes. In the following sections, decentralized process modeling and instance tracking is applied for this use case.

2.3 Decentralized Consensus Through Blockchains

When decentralized organizations are based on a blockchain, the organizational system itself is decentralized such that distributed components in the form of peers utilize consensus for distributed coordination. Decentralized consensus provided by blockchains is therefore the final consideration for decentralized organizations.

Blockchain systems establish consensus on integrity-secured, globally available, and non-repudiable transactions among the distributed and a priori unknown participants of a peer-to-peer network (Buterin 2013; Nakamoto 2008; Wood 2019). Such a system can be described in terms of the data structure of ordered blocks, the infrastructure of the peer-to-peer network and the consensus protocol (Härer 2019, p. 143 ff.). To be decentralized, this system must not only avoid concentration of the data structure, but also of the peer-to-peer network that executes the consensus protocol. This is the case if there are distributed system components in the form of peers that realize distributed coordination by carrying out a verifiable protocol. The protocol involves here the creation of blocks through mining and the rules determining their validity. The possibility for peers to inspect the blockchain and verify its correct operation by checking the consistency and integrity eliminates the need of a centralized party.

Decentralized consensus can be achieved through protocol functions such as voting mechanisms, or through smart contracts, which can be thought of as protocol extensions at run-time. Early on, the need for non-centralized governance has been recognized regarding the development

[1] https://git-scm.com/

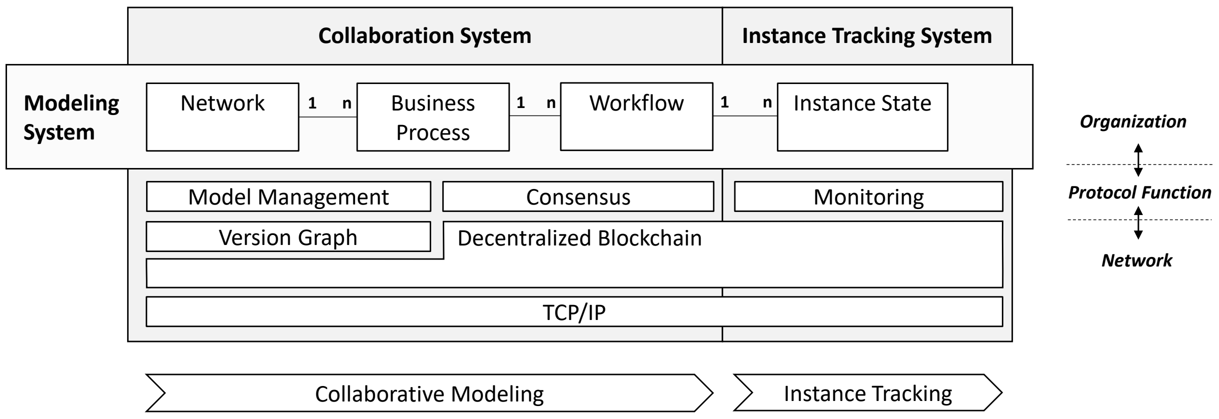

Figure 1: Architecture of the Approach

of blockchain protocols. In the Bitcoin blockchain, miners made decisions about backward-compatible protocol upgrades, i. e. soft forks, by setting a signaling bit in a newly created block to one of two values representing agreement or disagreement. Only if a previously defined share of miners agreed to an upgrade, source code for handling it was enabled. With smart contracts in Ethereum, this idea was taken further to decentralize the governance of organizations through smart contracts (Hsieh et al. 2018; Wood 2019). For a decentralized autonomous organization (DAO) (Mehar et al. 2019; Swan 2015), the idea of such a coordination was implemented in a smart contract, which was intended to handle funding and decision-making on investment autonomously. While this decentralized autonomous organization (DAO) failed due to a software bug which ultimately caused a backward-incompatible hard fork of the Ethereum protocol,[2] many software projects pursue versions of the idea. They include more elaborate forms of decentralized governance and organizational structures implemented in software (e. g. Aragon 2018).

Considering the manifestation of today's organizations in the physical world and the creation of value not only in digital and immaterial form, decentralized organizations might benefit from taking sociotechnical considerations into account. From the point-of-view of traditional companies and existing value creation networks, the design of a shared and trusted understanding of processes among parties with opposing interests might be supported by introducing decentralized consensus in their planning and execution.

3 Process Modeling in Decentralized Organizations

The aim of this approach is the design of decentralized organizations by modeling their planning of processes and by tracking their instances. Technologically, a decentralized blockchain is used to create a distributed system with distributed coordination realized by a blockchain protocol. In order to establish an appropriate architecture, the following requirements are defined corresponding to the research questions:

1. Representation of the decentralized system in terms of its peer network, processes, and instances.
2. Collaboration and reaching consensus on the aforementioned representations with the involvement of the distributed participants.
3. Tracking of instances in order to observe whether the execution occurs according to the agreed-upon representation.

These requirements concern both sociotechnical and technical aspects of a global system.

[2] https://blog.ethereum.org/2016/07/20/hard-fork-completed/

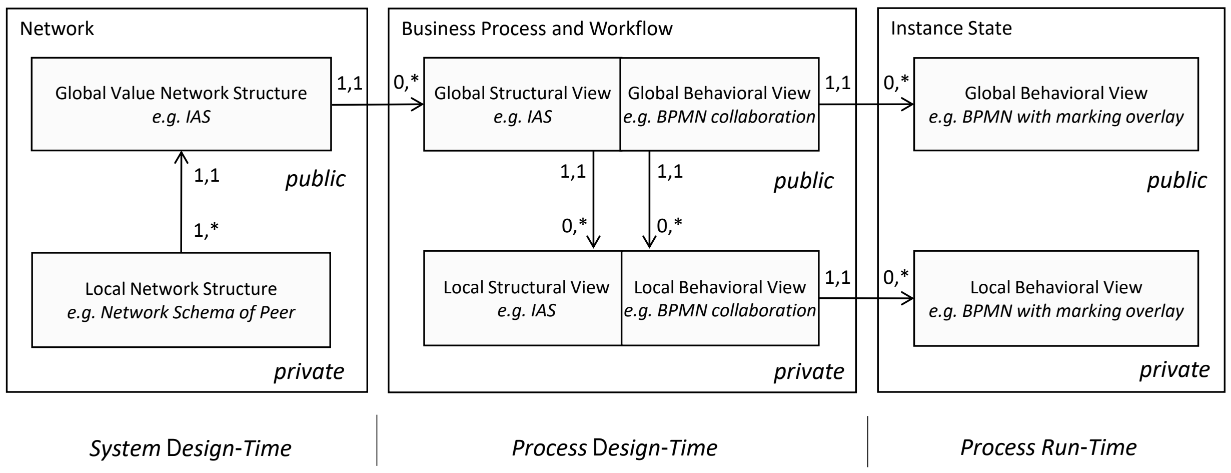

Figure 2: Model Artifacts

Fig. 1 shows an architecture designed to meet the requirements, where organizations are described by models in the uppermost layer. The modeling of the network describes the participants and relationships of the sociotechnical system from a structural point-of-view. Individual processes with their behavior, workflows, and their instances are established to the extent permitted by the chosen modeling languages.

From a technical point-of-view, the requirements can be considered functional requirements. A possible implementation is outlined by the subsystems of the architecture. Requirement 1 corresponds to the modeling system implementing modeling languages. Requirement 2 is addressed by the collaboration system containing functions for model management and consensus, executed in the same manner by all participants as a protocol. Model management is realized by a version graph for model versioning, and an underlying representation of all versions in a blockchain. Consensus relies on an agreement formed over a smart contract on the blockchain. In this fashion, the blockchain adds above the internet protocols a layer for forming agreements by integrity-secured models. From requirement 3, the instance tracking system follows in order to monitor the state of instances during execution such that individual instance states are represented on the blockchain. Additional non-functional requirements taken into account in particular for the blockchain-based implementation are the time of executing blockchain transactions and the cost in terms of transaction fees (Härer and Fill 2019b).

3.1 Modeling System

For distributed participants to collaboratively develop a shared understanding of processes, common modeling languages are required. In this first sub-system of the architecture, languages are defined by a specification, commonly using metamodels, and the definition of file formats. In order to describe a decentralized and process-oriented organization by model-based abstractions, conceptual models for the network, process, and instance domains are utilized. The system can be described by the artifacts shown in Fig. 2. The procedure and example models are given in Fig. 3. Over the next sections on network, business process and workflow, and instance state, the procedure is indicated in the numbering of sub-sections.

3.1.1 Network

In a decentralized organization, the systems' development emerges from within the system, i. e. it originates from the peers of the network. The discovery of peers, similar to peer-to-peer protocols (Steinmetz and Wehrle 2005), requires the exchange of identifiers or locators for peers.

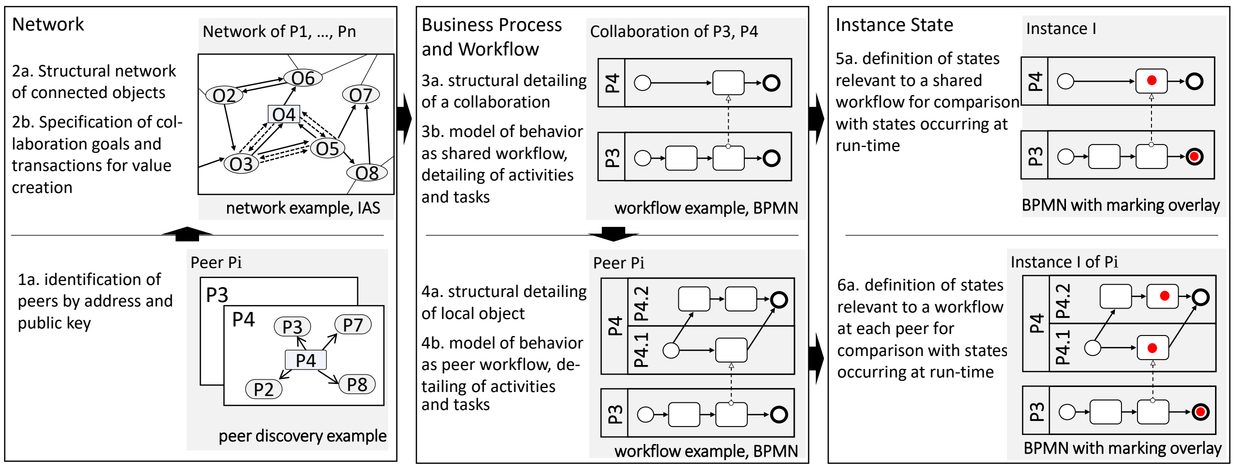
Network
Network of P1, ..., Pn
2a. Structural network of connected objects
2b. Specification of collaboration goals and transactions for value creation
network example, IAS
1a. identification of peers by address and public key
Peer Pi
peer discovery example
Business Process and Workflow
Collaboration of P3, P4
3a. structural detailing of a collaboration
3b. model of behavior as shared workflow, detailing of activities and tasks
workflow example, BPMN
4a. structural detailing of local object
4b. model of behavior as peer workflow, detailing of activities and tasks
Instance State
Instance I
5a. definition of states relevant to a shared workflow for comparison with states occurring at run-time
BPMN with marking overlay
Instance I of Pi
6a. definition of states relevant to a workflow at each peer for comparison with states occurring at run-time

Figure 3: *Procedure and Example Models*

1a. Peer Discovery

Before an organizational network can be formed, other actors need to be identified. From the perspective of a single actor, it needs to participate in the peer-to-peer network of a blockchain system in order to establish connections to other peers. In blockchain systems, addresses and corresponding public keys for signature verification are commonly used (Härer and Fill 2019a). For this reason, the local network of a peer, consisting of directly connected peers with their addresses and public keys, is described by the extension of the metamodel given in Fig. 4. A conforming model describes for a specific peer locally the known and connected remote peers.

2a. Structural Network

By joining the locally known peers on the address given for each peer, a global structure of connected objects is established. For each peer, at least one business object is created using the SOM interaction schema (IAS) (Ferstl and Sinz 2006). The resulting model contains the business objects of all peers globally. It serves as a basis for creating networks of business objects which engage in the organizational activities due to common goals for the creation of value. In the example in Fig. 5, the object *MA* denotes a manufacturer producing goods, with further connected objects of a middleman *MI* concerned with sourcing materials, two suppliers *SA* and *SB*, and a special carrier *SC* organizing deliveries (Fdhila et al. 2015). At this stage, only structural objects exist, however, the image taken from the implementation shows in addition the relations for collaboration goals and value creation.

2b. Collaboration Goals and Value Creation

Before the specification of processes, the global model of the network structure communicates all existing objects such that collaborations can be formed among objects with common goals. In order to form collaborations, the model is specified by time-continuous and transactional relations of the IAS. (1.) Common goals are specified and adjusted over time. The continuous exchange is modeled in goal and return relations. (2.) Transaction specify the value creation between objects (Ferstl and Sinz 2006). The example of the network model in Fig. 5 displays goals that are mu-

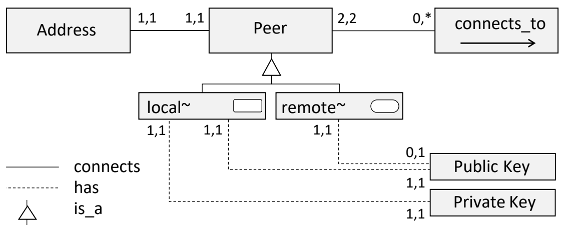

Figure 4: *Local Network Metamodel (Syntax: Sinz 1996)*

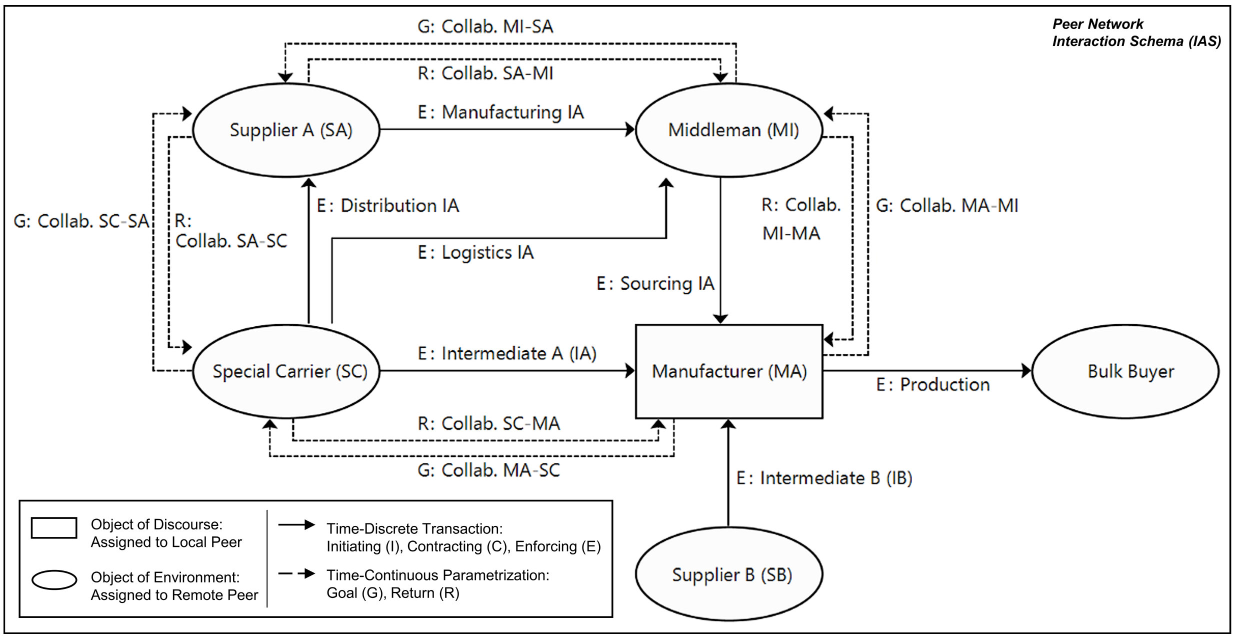

Figure 5: Global Value Network Model

tually defined between collaborating objects. For example, the goal of material sourcing between *MA* and *MI* constitutes the collaboration and leads to the specification of the transaction from *MI* to *MA* that sources an intermediate A (IA).

3.1.2 Business Process and Workflow

The specification of the process-oriented system out of the known structure given by the model of the network follows. According to the goals of a set of collaborating objects, they are decomposed together with the transactions connecting them in the manner suggested by SOM (Ferstl and Sinz 2006).

3a. Global Structural View

For any number of collaborating objects, the structural view of the collaboration is decomposed in terms of their transactions and possibly further objects. With the example of SOM, any transaction representing value creation may be decomposed into multiple transactions according to the three negotiation phases initiating (I), contracting (C), and enforcing (E). Since E is always present, the patterns I, C, E or C, E or E can occur, however, each transaction might in addition be specialized or decomposed further. In the upper part of Fig. 6, the global view of the manufacturing process is displayed. On the global level, the objects forming a collaboration are denoted objects of discourse. Transactions are decomposed, such as order (C) and manufacturing (E) between *MA* and *Bulk Buyer*. The manufacturing transaction itself is not visible due to further specialization into transactions for production start, production end, and delivery.

3b. Global Behavioral View

A behavioral view determines the individual tasks of the process, which can be detailed into workflow activities. Here, a BPMN collaboration diagram is derived as a shared workflow. Metamodel-based transformations can be utilized at this point. The transformation from SOM to BPMN (Pütz and Sinz 2010) leads to one pool per object with message flows resulting from the transactions. Further activities and other flow objects may be added with parameters for the specification of an executable workflow. The global behavioral view is not shown in Fig. 6 due to space constraints, however, the model resulting from the transformation of the structural view, displayed in the upper part, is the BPMN collaboration shown below without the activities and gateways from the decomposition of *MA*.

4a. Local Structural View
The decomposition approach can be utilized to specify the local view of each peer. Subsequently, the hierarchical decomposition of objects and transactions details the structure of the public process in a private process (van der Aalst and Weske 2001). The compositions of objects and transactions form a tree. The local view of a private process is derived using the complete trees. From the hierarchies, the public process is derived by pruning all nodes below the last public decomposition level of the object tree and all transactions connected to them. The model in the lower part of Fig. 6 shows the local decomposition of *MI* and *MA*. The local view of *MA* contains the objects *Distribution* and *Production* as decomposition products of *MA*.

4b. Local Behavioral View
A local behavioral view can be detailed to the level of a peer workflow. With the assumption of a hierarchical decomposition, activities of the shared workflow are specified according to the previously defined structure. The behavior resulting from the private workflow must match the publicly observable behavior of the shared interorganizational workflow. When using petri-nets derived from the shared workflow, the existing structure in effect constructs a hierarchical petri-net according to the decompositions made within the process. Fig. 6 describes in the lower part between *MI* and *MA* a collaboration where message flows show the exchange of a sourcing inquiry for an intermediate A (IA) and the return of corresponding information about arrangements made.

3.1.3 Instance State

With a growing distribution of process participants, the dependable definition of instance states reached during execution is required in order to track the progress and success of planned processes.

5a. Definition of Global Instance States
Instance state models record an execution state on the basis of the existing workflow definition. With the assumption of unambiguous operational semantics for the example of BPMN, an instance state model may be specified using the shared workflow model with a marking overlay. An alternative is a derivation of petri-nets. Any number of states define the beginning, intermediate, and final states of a workflow. The states relevant for a workflow are later applied to a hash function in order to find matching states at run-time. In the implementation, BPMN is used with a marking overlay showing the state of the execution. Fig. 10 gives an example of this approach where the execution state of one instance *I1* can be seen.

6a. Definition of Local Instance States
The local peer workflow serves as a basis for the definition of states. According to the previous paragraph, locally relevant states are captured in instance state models. That is, a marking overlay is applied to an existing model of a local behavioral view. Due to the known decomposition of the globally defined process and its BPMN representation, all objects on the local level of a particular peer can be traced back to their parent objects. Fig. 10 shows *I1* locally for two peers with their respective local instance states. The local decomposition of the peer *Manufacturer* (*MA*) contains two activities resulting from the decomposition of *MA*.

3.2 Collaboration System

The collaboration system allows for the creation of models by multiple participants with a mechanism for creating consensus among them. In order for participants to make decisions through model-based abstractions, the integrity and non-repudiation properties of blockchain systems are utilized, i. e. both properties need to be independently verifiable by the participants.

3.2.1 Model Management

When applying these properties to the management of models in a decentralized setting, it is insufficient to construct a model repository only

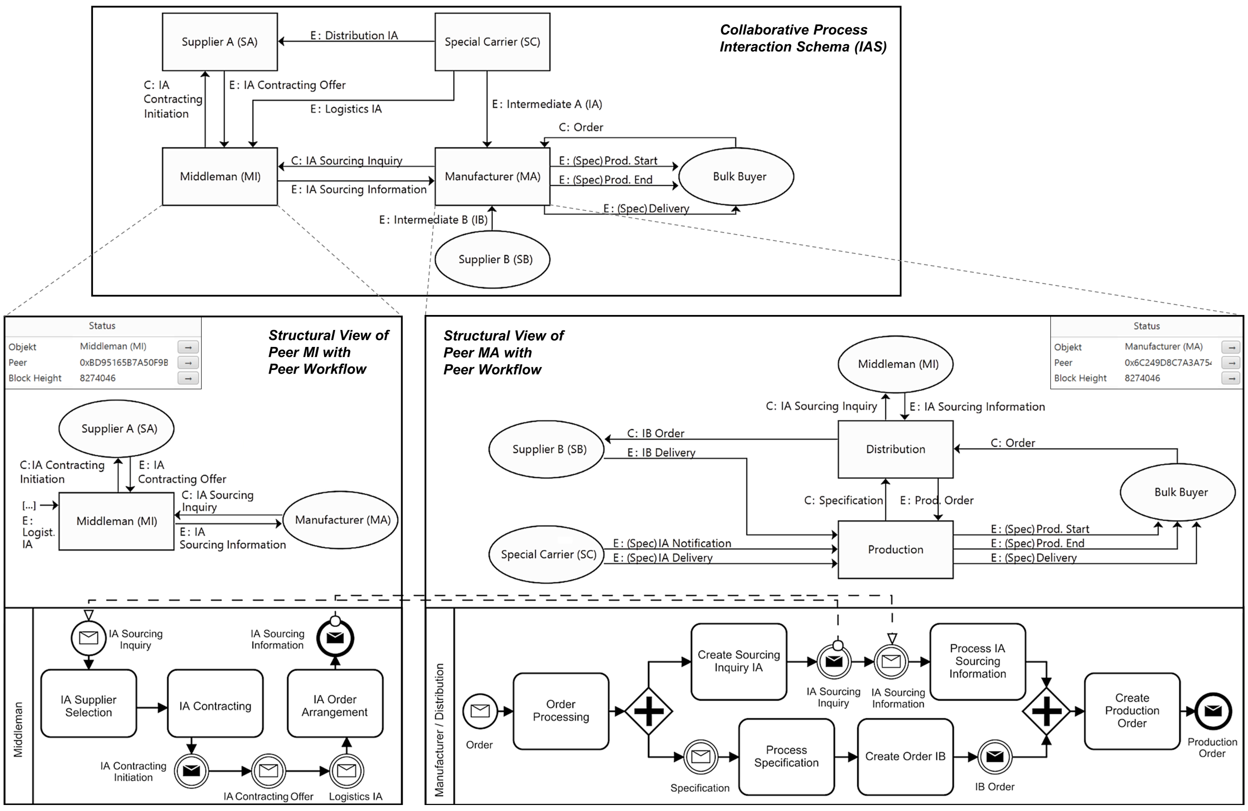

Figure 6: Business Process and Workflow Models

on the basis of a smart contract in a public blockchain with limited scalability. Due to transaction size and cost, the storage in smart contracts needs to be limited to the data required for validating the properties integrity and non-repudiation. Asynchronous versioning can be utilized as a basis,

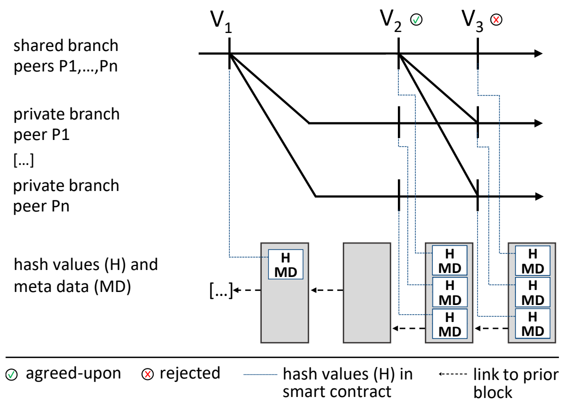

Figure 7: Versioning Secured by a Blockchain

as it already presents a way of managing models. Fig. 7 shows the combination of a version control system with a smart contract used for transactional modeling. For the differentiation of public and private models, related branches with global and local availability are introduced. In this asynchronous communication mode, reaching agreements is based on explicitly defined global commit operations.

3.2.2 Agreement on the Modeling System

In order to satisfy the requirement of non-centrally controlled decision-making through the integrity and non-repudiation properties, version control is extended with a smart contract.

Integrity is determined by a hash function, applied to the modeling system. The resulting value is stored for each version in the smart contract and verified every time the modeling system has been retrieved from version control. Non-repudiation is a consequence of the digital signatures, which are

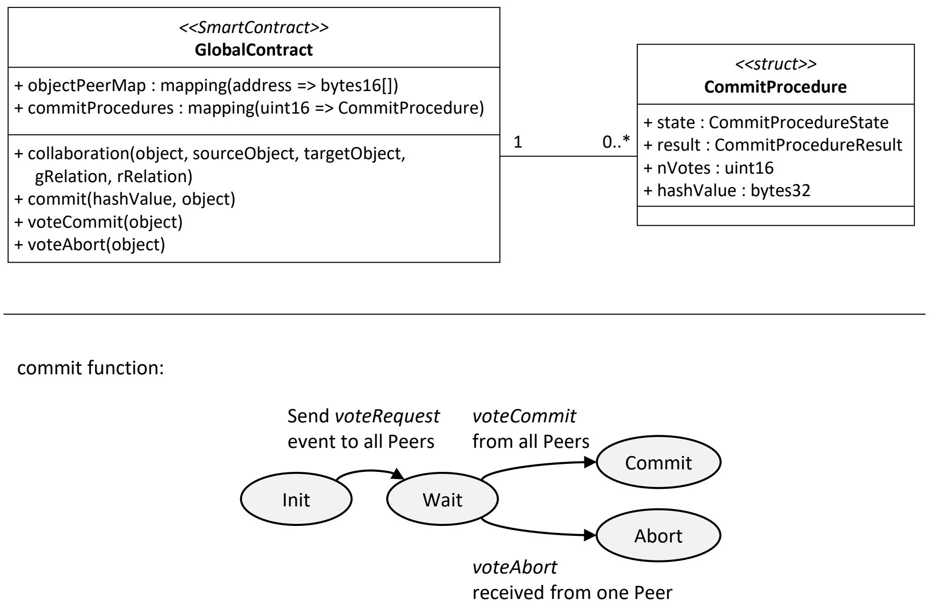
<<SmartContract>>
GlobalContract
+ objectPeerMap : mapping(address => bytes16[])
+ commitProcedures : mapping(uint16 => CommitProcedure)
+ collaboration(object, sourceObject, targetObject, gRelation, rRelation)
+ commit(hashValue, object)
+ voteCommit(object)
+ voteAbort(object)
1
0..*
<<struct>>
CommitProcedure
+ state : CommitProcedureState
+ result : CommitProcedureResult
+ nVotes : uint16
+ hashValue : bytes32
commit function:
Send voteRequest event to all Peers
voteCommit from all Peers
Init
Wait
Commit
Abort
voteAbort received from one Peer

Figure 8: Smart Contract

produced by the author of every commit through their private key. The corresponding public key, shared through the peer network model, allows for the validation of signatures by all participants at a later point in time.

Fig. 8 shows the class diagram of the agreement functions of the global smart contract. In order to form a collaboration among peers, a related function is invoked by a peer. Here, the references to the previously defined goal and return relations need to be given. The mutual definition of goals can be restricted further by setting appropriate types and restrictions on ordering by separate functions.

A distinction between the commit operations for local models and global models of the shared process is made. Global commit operations made to the public branch modify the public process and its workflow. Therefore, the global commit is carried out as a distributed commit implementing a voting scheme. In contrast to a distributed system, the commit procedure can be based on the blockchain protocol, which establishes a consensus layer below the commit scheme used. In order to implement a decision-making mechanism in this fashion, votes are represented by the outcome of a two-phase commit in a smart contract. Here, an agreement of all participants is required in order to reach consensus. In the same way, more elaborate voting schemes can also be implemented.

The two-phase commit carried out by the smart contract in conjunction with the version graph is described in Fig. 9. Transactions concern the peer locally as well as globally. The local transaction (LT) consists of modeling operations, carried out until a syntactically and semantically correct state is reached. The global transaction (GT) is initiated by a commit function, which calls the smart contracts' commit function and creates a new version on the version graph. The smart contract initiates the commit after confirmation on the blockchain. Any peer of the collaboration is able to vote according to the version fetched from the shared branch. The voteRequest event emitted by the smart contract carries the object identifier and hash value of the commit. In case of their successful validation, the voteCommit or voteAbort function is invoked for voting. In the

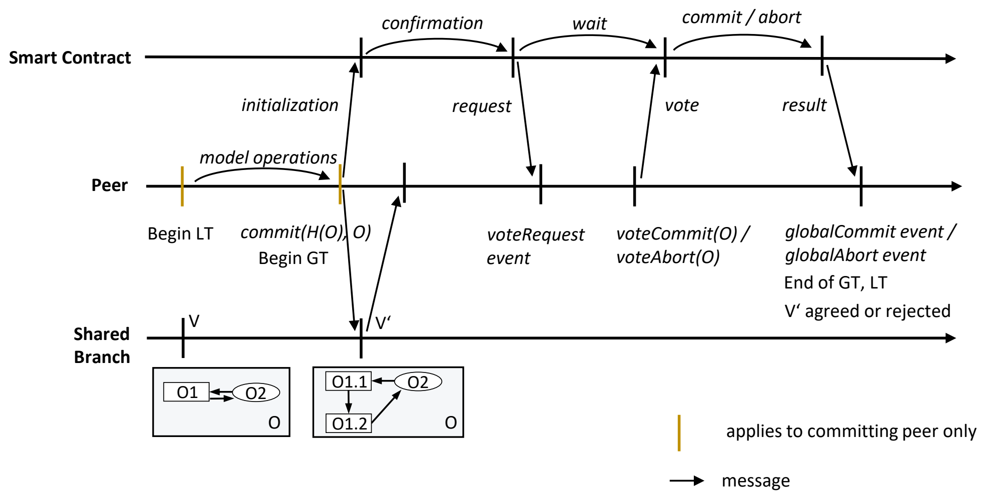

Figure 9: Two-Phase Commit

other case, the commit is aborted. The global and local commit end, when either one voteAbort call is processed by the smart contract, or when the voteCommit calls from the addresses of all registered peers of a collaboration are completed. The version is then agreed or rejected.

3.3 Instance Tracking System

In order to coordinate the progression of processes among distributed peers, instance tracking is required. The observation of the instance states of the shared workflow must match the behavior specified by its model. The instance state models defined on the basis of the workflow models represent check-points which are of relevance to the workflow execution. Models of the initial, intermediate, and final states provide a reference for determining a successful, failed, or exceptional outcome. Using the infrastructure for the global storage of models established through the collaboration system, the commit of instance state models can have a binding character. Similar to a signed transaction for the transfer of virtual currency, the binding transmission of an instance state containing one or more tokens can occur. Similar to the validation of the transaction history in digital currencies, the record of prior states is a protocol log of the execution. Any peer determines the validity of recorded instance state models. I. e., an instance state model is valid if it is syntactically correct and it is reached through valid transitions defined by the shared workflow under the assumption of petri-net semantics. Else, the model is discarded. After a successful validation, it is determined whether or not an instance state model is in a previously specified beginning, intermediate, or final state. Any state specified as part of the modeling system is identified by its hash value through the collaboration system in the manner described before. For comparing specified states with actual states, the hash values are compared. Fig. 10 shows an example.

3.4 Implementation Technologies

This section discusses the implementation technologies chosen for the developed prototype.[3] The aim of the prototype is to show the feasibility of implementing the modeling of processes in decentralized organizations. On the basis of the public internet, Ethereum is utilized as an open and permissionless blockchain system. In order to provide blockchain properties such as integrity, non-repudiation and immutability, a node software which fully downloads and validates all blocks is required. Here, parity is used in a configuration

[3] The software and corresponding smart contracts are available online at: https://github.com/fhaer/Process-Modeling-in-Decentralized-Organizations

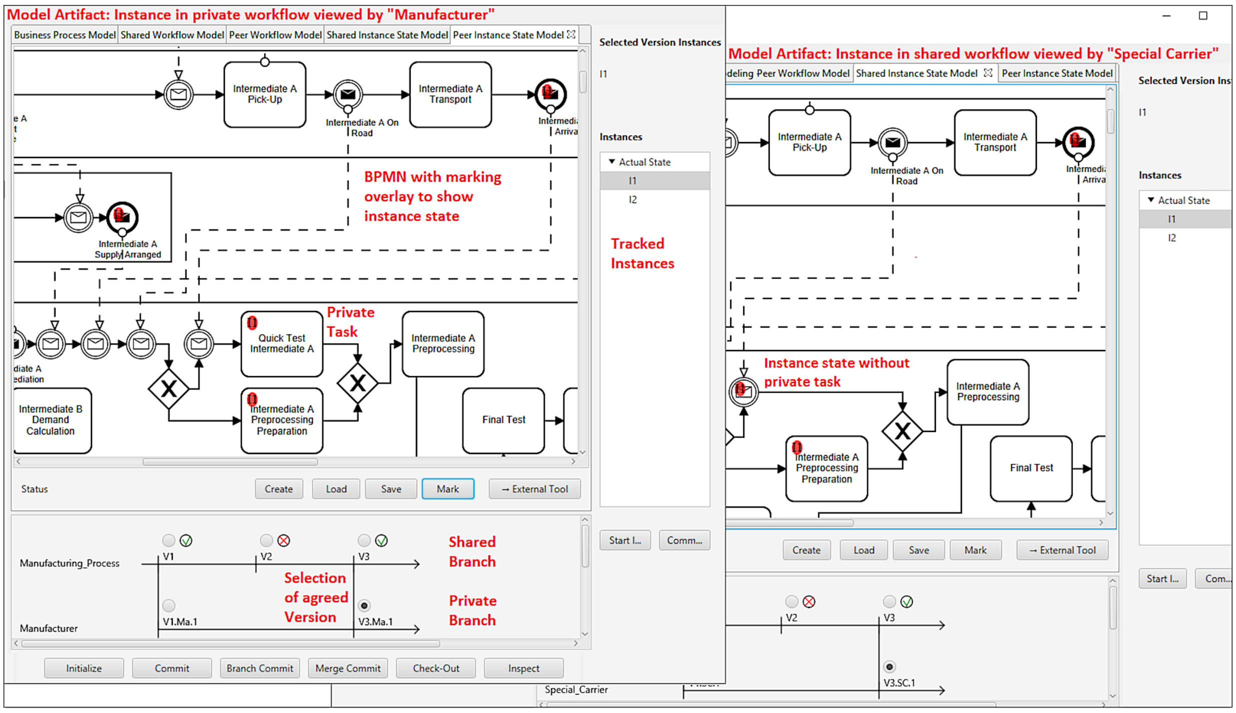
Model Artifact: Instance in private workflow viewed by "Manufacturer"
Business Process Model
Shared Workflow Model
Peer Workflow Model
Shared Instance State Model
Peer Instance State Model
Selected Version Instances
Intermediate A Pick-Up
Intermediate A Transport
Intermediate A On Road
BPMN with marking overlay to show instance state
Intermediate A Supply Arranged
Tracked Instances
Instances
Actual State
I1
I2
Private Task
Quick Test Intermediate A
Intermediate A Preprocessing
Intermediate B Demand Calculation
Intermediate A Preprocessing Preparation
Final Test
Status
Create
Load
Save
Mark
→ External Tool
Manufacturing_Process
Manufacturer
V1
V2
V3
V1.Ma.1
V3.Ma.1
Selection of agreed Version
Shared Branch
Private Branch
Start I...
Comm...
Initialize
Commit
Branch Commit
Merge Commit
Check-Out
Inspect
Model Artifact: Instance in shared workflow viewed by "Special Carrier"
Instance state without private task
Special_Carrier
V3.SC.1

Figure 10: Software Instances at Two Peers (Härer 2018)

where the entirety of blockchain transactions is validated and archived locally with only recent state representations being present in a cache (Härer and Fill 2019b). Deployed on Ethereum[4] is a smart contract operating globally for the registration of peers with their addresses, the storage of integrity data for global commits, and for voting processes in the form of the two-phase commit. A second object-specific smart contract handles the model versioning and execution for individual peers. For the object *Manufacturer* in the process models presented, the contract is deployed on Ethereum.[5] Within the program, the sending of transactions to the contracts and the querying of their state variables are realized through the API web3.js in the web3j library for Java.[6] It is invoked through the prototype written in JavaFX. The software exposes the agreement functionality to users and allows for model management through the Git version control system. For the creation of design-time models, the prototype contains modeling editors, however, external tools such as ADOxx may be used in addition. As an example, Fig. 10 shows a screenshot of a first prototype illustrating instance state models at two distinct peers in a collaborative process scenario (Härer 2018). Fig. 6 shows the process model editor with an excerpt of the interface of the current version of the prototype.

4 Discussion

The information system of decentralized organizations needs to have the characteristics of a decentralized system. Given the assumed notion of a decentralized system, it is required to have distributed system components which coordinate their actions by a protocol that they execute and verify. On an organizational level, individual components are represented in their structure for them to collaboratively design the behavior of processes. The coordination of individual system

[4] Address: 0xD38CF1FDAB83DAE505224A1F8DC264E3FB85C24E, see e. g. https://etherscan.io/address/<address>
[5] Address: 0x6c249d8c7a3a75419d1fe2dcfb0644fb5ea9a60a
[6] http://web3j.io/

components is determined by process models and observed in instance models when executed.

Thus, the representation, the formation of consensus, and the observation of run-time behavior are addressed by this approach in accordance with the research questions posed. This is reflected in the architecture of the approach, where the representation results out of the modeling system, the formation of consensus is the result of the collaboration system, and instance tracking allows for the observation of run-time behavior.

A distributed system under distributed coordination is mainly established by the hierarchical decomposition of the information system components and agreements based on a voting mechanism implemented in a smart contract. In this way, decision-making is based on binding models handled by self-organized peers. The decentralized planning can be supported due to the transparent access to conceptual models, to the degree permitted by modeling languages. In principle, after the identification of peers, the decision-making on the network and the subsequent decomposition of public and private processes is limited to the voting mechanism described. In its current realization, the mechanism requires a unanimous vote. Limiting technical factors concern the properties of the underlying blockchain system. Transaction cost and confirmation times need to be considered for every commit of the modeling system. Since blockchain transactions only contain hash values, object identifiers, and goal relations, their size is comparatively small compared to the data stored in the version control system. The transaction input data size for a modeling system commit on Ethereum is 500-600 bit, which currently results in transaction cost on the order of ten USD cents with transaction confirmation times being on the order of tens of seconds (Härer and Fill 2019b).

5 Conclusion

The process-oriented design and instantiation of decentralized organizations can be achieved by their representation in conceptual models, by a mechanism providing for consensus on the model-based design, and by the observation of distributed instances at run-time. Conceptual modeling is suitable for representing the planning and execution of processes in decentralized organizations.

Blockchain consensus can be applied to reach a consistent representation of processes. In effect, they can be globally validated as part of the blockchain data structure consisting of integrity-secured and non-repudiable transactions.

Organizations and processes might be designed with decentralized decision-making, e. g. among a network of suppliers as illustrated by the models in this paper, or for the creation of digital products by partially or fully autonomous decentralized organizations implemented in smart contracts. Examples of the latter category are markets for tokenized assets, decentralized exchanges, and platforms selling cloud computing resources for digital currencies. The examples have in common that they operate without centralized control and coordination of third parties, such that the design of the structure, processes, and execution are not in the hands of a single actor but are shared by all parties involved. The primary benefits expected from such a decentralized system are fewer intermediaries and trusted third parties due to the direct involvement and control of the parties involved.

At this point, the approach focuses on the feasibility of decentralized coordination through conceptual modeling. While the implemented voting mechanism shows this possibility, more complex governance mechanisms can be realized. In future research, limitations of technical nature of governance and blockchain systems as well as sociotechnical considerations of decentralization need to be addressed for information systems to become decentralized.

References

Altmanninger K., Seidl M., Wimmer M. (2009) A Survey on Model Versioning Approaches. In: International Journal of Web Information Systems 5(3), pp. 271–304

Aragon (2018) Aragon https://aragon.org/

Bleicher K. (1991) Organisation: Strategien - Strukturen - Kulturen, 2nd edition. Gabler

Brosch P., Kappel G., Langer P., Seidl M., Wieland K., Wimmer M. (2012) An Introduction to Model Versioning. In: Proceedings of the 12th International Conference on Formal Methods for the Design of Computer, Communication, and Software Systems: Formal Methods for Model-Driven Engineering. SFM'12. Springer, pp. 336–398

Buterin V. (2013) Ethereum: The Ultimate Smart Contract and Decentralized Application Platform http://web.archive.org/web/20131228111141/http://vbuterin.com/ethereum.html

Camunda (2018) The Camunda BPM Manual, BPM Platform 7.10 https://docs.camunda.org/manual/7.10/

Christie A. A., Joye M. P., Watts R. L. (2003) Decentralization of the Firm: Theory and Evidence. In: Journal of Corporate Finance 9(1), pp. 3–36

Di Ciccio C., Cecconi A., Dumas M., García-Bañuelos L., López-Pintado O., Lu Q., Mendling J., Ponomarev A., Binh Tran A., Weber I. (2019) Blockchain Support for Collaborative Business Processes. In: Informatik Spektrum 42(3)

Dupont Q. (2017) Experiments in Algorithmic Governance: A History and Ethnography of "The DAO", a Failed Decentralized Autonomous Organization. In: Bitcoin and Beyond: Cryptocurrencies, Blockchains and Global Governance. Routledge

Evermann J., Kim H. (2019) Workflow Management on the Blockchain – Implications and Recommendations. Online at https://arxiv.org/abs/1904.01004

Fdhila W., Rinderle-Ma S., Knuplesch D., Reichert M. (2015) Change and Compliance in Collaborative Processes. In: 2015 IEEE International Conference on Services Computing. IEEE, pp. 162–169

Ferstl O. K., Sinz E. J. (2006) Modeling of Business Systems Using SOM. In: Bernus P., Mertins K., Schmidt G. (eds.) Handbook on Architectures of Information Systems. Springer, pp. 347–367

Fill H.-G. (2019) Applying the Concept of Knowledge Blockchains to Ontologies. In: Proceedings of the AAAI 2019 Spring Symposium on Combining Machine Learning with Knowledge Engineering (AAAI-MAKE 2019). Stanford University

Fill H.-G., Härer F. (2018) Knowledge Blockchains: Applying Blockchain Technologies to Enterprise Modeling. In: 51st Hawaii International Conference on System Sciences (HICSS-51), pp. 4045–4054

Fill H.-G., Karagiannis D. (2013) On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. In: Enterprise Modelling and Information Systems Architectures (EMISAJ) 8(1), pp. 4–25

Haarmann S., Batoulis K., Nikaj A., Weske M. (2018) DMN Decision Execution on the Ethereum Blockchain. In: Krogstie J., Reijers H. A. (eds.) Advanced Information Systems Engineering Vol. 10816. Springer, pp. 327–341

Härer F. (2018) Decentralized Business Process Modeling and Instance Tracking Secured By a Blockchain. In: Proceedings of the 26th European Conference on Information Systems (ECIS)

Härer F. (2019) Integrierte Entwicklung Und Ausführung von Prozessen in Dezentralen Organisationen. Ein Vorschlag Auf Basis Der Blockchain. Schriften Aus Der Fakultät Wirtschaftsinformatik Und Angewandte Informatik Der Otto-Friedrich-Universität Bamberg 39. University of Bamberg Press

Härer F., Fill H.-G. (2019a) A Comparison of Approaches for Visualizing Blockchains and Smart Contracts. In: Jusletter IT Weblaw February 2019

Härer F., Fill H.-G. (2019b) Decentralized Attestation of Conceptual Models Using the Ethereum Blockchain. In: 21st IEEE International Conference on Business Informatics (CBI 2019)

Hsieh Y.-Y., Vergne J.-P., Anderson P., Lakhani K., Reitzig M. (2018) Bitcoin and the Rise of Decentralized Autonomous Organizations. In: Journal of Organization Design 7(1), p. 14

Karagiannis D., Mayr H. C., Mylopoulos J. (2016) Domain-Specific Conceptual Modeling. Springer

Kelly S., Tolvanen J.-P. (2018) Collaborative Creation and Versioning of Modeling Languages with MetaEdit+. In: Proceedings of the 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion Proceedings (MODELS 18). ACM Press, pp. 37–41

Ladleif J., Weske M., Weber I. (2019) Modeling and Enforcing Blockchain-Based Choreographies. In: Hildebrandt T., van Dongen B. F., Röglinger M., Mendling J. (eds.) Business Process Management. Lecture Notes in Computer Science. Springer, pp. 69–85

Liu C., Li Q., Zhao X. (2009) Challenges and Opportunities in Collaborative Business Process Management: Overview of Recent Advances and Introduction to the Special Issue. In: Information Systems Frontiers 11(3), pp. 201–209

López-Pintado O., Garćıa-Bañuelos L., Dumas M., Weber I., Ponomarev A. (2019) Caterpillar: A Business Process Execution Engine on the Ethereum Blockchain. In: Software: Practice and Experience 49(7)

Mehar M. I., Shier C. L., Giambattista A., Gong E., Fletcher G., Sanayhie R., Kim H. M., Laskowski M. (2019) Understanding a Revolutionary and Flawed Grand Experiment in Blockchain: The DAO Attack. In: Journal of Cases on Information Technology (JCIT) 21(1), pp. 19–32

Mendling J., Weber I., van der Aalst W., Vom Brocke J., Cabanillas C., Daniel F., Debois S., Di Ciccio C., Dumas M., Dustdar S., Gal A., Garc-Bañuelos L., Governatori G., Hull R., La Rosa M., Leopold H., Leymann F., Recker J., Reichert M., Reijers H. A., Rinderle-Ma S., Rogge-Solti A., Rosemann M., Schulte S., Singh M. P., Slaats T., Staples M., Weber B., Weidlich M., Weske M., Xu X., Zhu L. (2018) Blockchains for Business Process Management - Challenges and Opportunities. In: ACM Transactions on Management Information Systems 9(1), pp. 1–16

Nakamoto S. (2008) Bitcoin: A Peer-to-Peer Electronic Cash System https://bitcoin.org/bitcoin.pdf

Nicolaescu P., Rosenstengel M., Derntl M., Klamma R., Jarke M. (2018) Near Real-Time Collaborative Modeling for View-Based Web Information Systems Engineering. In: Information Systems Information Systems Engineering: Selected Papers from CAiSE 2016 74, pp. 23–39

Object Management Group (2014) Business Process Model and Notation (BPMN), Version 2.0.2, formal/2013-12-09 http://www.omg.org/spec/BPMN/2.0.2

Piller F. T., Möslein K., Ihle C. C., Reichwald R. (2017) Interaktive Wertschöpfung kompakt: Open Innovation, Individualisierung und neue Formen der Arbeitsteilung. Springer

Pütz C., Sinz E. J. (2010) Model-Driven Derivation of BPMN Workflow Schemata from SOM Business Process Models. In: Enterprise Modelling and Information Systems Architectures (EMISAJ) 5(2), pp. 57–72

Scott B., Loonam J., Kumar V. (2017) Exploring the Rise of Blockchain Technology: Towards Distributed Collaborative Organizations. In: Strategic Change 26(5), pp. 423–428

Simon H. A., Guetzkow H., Kozmetsky G., Tyndall G. (1954) Centralization vs. Decentralization in Organizing the Controller's Department.: A Research Study and Report Prepared for Controllership Foundation, Inc.. Series A-1 Controllership: Contracts, Organization. Report, No. 4. Controllership Foundation

Sinz E. J. (1996) Ansätze Zur Fachlichen Modellierung Betrieblicher Informationssysteme. Entwicklung, Aktueller Stand Und Trends. In: Heilmann H., Heinrich L., Roithmayr F. (eds.) Information Engineering. Oldenbourg

Sinz E. J. (2019) On the Evolution of Methods for Conceptual Information Systems Modeling. In: Bergener K., Räckers M., Stein A. (eds.) The Art of Structuring: Bridging the Gap Between Information Systems Research and Practice. Springer, pp. 137–144

Steinmetz R., Wehrle K. (2005) Peer-to-Peer Systems and Applications. Springer

Swan M. (2015) Blockchain: Blueprint for a New Economy, First edition. O'Reilly

Tran A. B., Lu Q., Weber I. (2018) Lorikeet: A Model-Driven Engineering Tool for Blockchain-Based Business Process Execution and Asset Management. In: 16th International Conference on Business Process Management (BPM 2018)

van der Aalst W., Weske M. (2001) The P2P Approach to Interorganizational Workflows. In: Dittrich K. R., Geppert A., Norrie M. C. (eds.) Advanced Information Systems Engineering. Lecture Notes in Computer Science. Springer, pp. 140–156

Weber I., Xu X., Riveret R., Governatori G., Ponomarev A., Mendling J. (2016) Untrusted Business Process Monitoring and Execution Using Blockchain. In: La Rosa M., Loos P., Pastor O. (eds.) 14th International Conference, Business Process Management (BPM 2016), pp. 329–347

Wood G. (2019) Ethereum: A Secure Decentralised Generalised Transaction Ledger https://ethereum.github.io/yellowpaper/paper.pdf

Xu X., Weber I., Staples M., Zhu L., Bosch J., Bass L., Pautasso C., Rimba P. (2017) A Taxonomy of Blockchain-Based Systems for Architecture Design. In: 2017 IEEE International Conference on Software Architecture (ICSA). IEEE, pp. 243–252