Negotiating choreography models based on distributed ledger technologies

Christian Friedow, Maximilian Völker, and Stephan Haarmann

Hasso Plattner Institute, University of Potsdam, Germany,
stephan.haarmann@hpi.de

Abstract. Key words: Business Processes, Choreography, BPMN

1 Introduction

Choreography diagrams can be seen as a type of contract between the participating parties [1]. Whilst those choreographies can already be executed in a more secure way using the blockchain as a distributed and tamper-proof interaction layer, how about the process of "negotiating" the model itself? This process also involves all parties and each of them should need to confirm the model before it can be deployed. But who is responsible for modeling and storing the model and its intermediate stages? Who tracks that everyone really agreed to the current model? So the process of modeling and negotiating still requires trust: Trust in the modeling party, not to secretly change the diagram. Trust in the deployment, not to put an unseen model in operation. Trust in an organizer, to mediate and to keep track of the most recent and accepted version.

In this paper, we propose a way to model and negotiate choreography diagrams in a traceable, transparent and secure way. The approach is built on distributed ledger technologies to store and orchestrate the collaboration. It ensures a consistent modeling process and that each change to the model is reviewed by all concerned collaborators. As the approach also includes a graphical interface, all participants can model and propose changes. Using the blockchain technology, all proposals, changes and reviews are stored in a tamper-proof and accessible way, without the need of a trusted third party. All participants also store the history locally in their own synchronized ledger, the "ground truth", no party is able to introduce unseen changes to the model.

HERE COMES THE OUTLINE OF THE PAPER, CHAPTER 2 INTRODUCES RELATED WORK, CHAPTER 3 DESCRIBES THE APPROACH, CHAPTER 4 OUR PROTOTYPICAL IMPLEMENTATION, CHAPTER 5 DRAWBACKS, FUTURE WORK AND SO ON.

2 Related Work

– Execution of processes on blockchain

– Process binary trees
– Choreography diagrams as contracts

According to [1], Choreography diagrams can be seen as contracts between the participating organizations. Choreographies provide a way to define how participants interact and in which order information (messages) is exchanged.

3 Approach

In the following, we present an approach for modeling choreography diagrams in a secure, tamper-proof and collaborative way. This enables a more transparent, traceable and reliable collaboration or "negotiation" process and paves the way for the use of choreography diagrams in more formal and strict use cases.

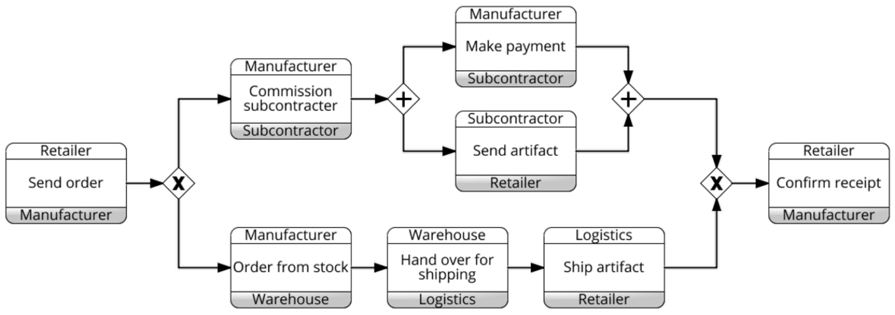

Fig. 1. An exemplary choreography diagram

3.1 Collaborative negotiation platform

To kick off the negotiation process a model, exemplary shown in fig. 1, has to be created first. After adding other modellers to collaborate with, the process of a negotiation round, shown in fig. 2, is applied to every change made to the model. To start a negotiation round a modeler proposes an altered version of the model. Since working with a blockchain is sensitive to space and computation power limitations the model is converted into a binary tree first. Only the binary tree and a hash of the altered model are committed to the chain in order to save space. Furthermore, analytical operations can be performed on-chain with using less computation power when working with a binary tree instead of the textual representation of the model. Section 3.2 covers the conversion of a model into a binary tree in detail. After the change as has been proposed successfully, the next step is to determine the list of reviewers needed to examine the change. This is done utilizing the concept of trust levels, explained in section 3.3, and based on analysis of the binary tree. Furthermore, the proposer can add additional reviewers manually before starting the review process. To complete the review process, each reviewer is asked to revise the altered version of the model. Therefore one has either the option to approve the proposed change or to create a counterproposal. A counterproposal is an altered version of the model based on the proposed change, which starts a new negotiation round on its own. To avoid the issue of merging multiple versions of a model only one proposal can exist at a time. That is why, submitting a counterproposal will automatically archive and lock the proposal it was based on. To successfully complete the negotiation process and to agree on a new model all participating reviewers need to accept the proposed change.

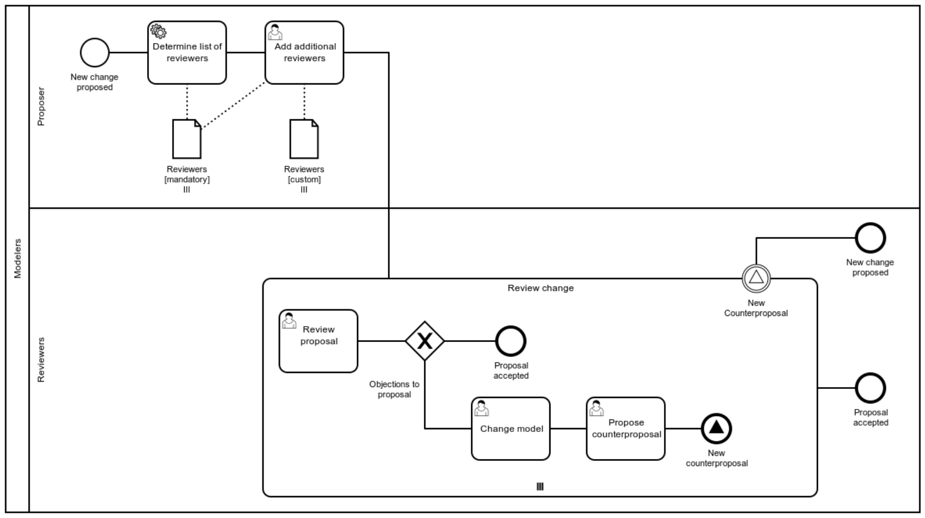

Fig. 2. Negotiation process overview

3.2 Model to binary tree conversion

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

3.3 Trust levels

With an increasing number of parties involved in the choreography, the review process can become tedious if every party wants to check every change. Therefore we propose a way to mitigate this effort by introducing a per-collaboration setting, the "trust level". If we look at negotiations and collaborations, each group will have a different attitude to each other: A few long-standing, well acquainted business partners might start modeling with a higher degree of trust, than a large group of new partners would. As a collaborator, this means that if a well-known partner changes a part of the choreography (or contract) that does not concern me in any way, I would not have to be part of the reviewing group, because I "trust" him/her. On the other hand, if my co-participants are new to me, I want to review their changes, even if they do not affect parts concerning me.

Path based trust levels The first approach for those "trust levels", is a setting per diagram. When beginning a new choreography negotiation, the collaborators agree on a specific level. This level will then influence the algorithm determining the list of reviewers needed for a certain proposal or change. In our approach we provide the following five gradations:

1. **All review:** Every participant in the model has to review and approve a change in order to integrate it into the accepted model.
2. **Path review:** Collaborators that participate in an activity that might be executed before, in parallel, or after the change have to approve the proposal. When using this level, a participant will not need to review a change if he/she is only part of activities that cannot be executed in the same instance with the change. Referring to fig. 1, if the activity "Make payment" would have been changed, the participants *Warehouse* and *Logistics* would not need to give their approval, as their activities can never be executed in the same instance as the changed one.
3. **Subsequent review:** Participants of activities that might be executed in parallel or after the change have to review the proposal.
4. **Neighbourhood review:** This level already assumes a high level of trust amongst the collaborators but ensures that at least a few different parties have review the change. With this setting, participants of activities that are directly executed before, concurrently or after the change will have to approve the proposal.
5. **Direct review:** The highest level of trust, where only the participants of the changed activity have to review and approve the change.

Party-specific trust levels Setting the trust level per diagram might not do justice to every circumstance. In collaborations there can be subgroups trusting each other more than others. Therefore, the setting can also be introduced being set per-collaborator: When starting a new choreography negotiation, each participant can set the trust level regarding any other collaborator. Now, each "negotiation partner" can determine individually, whose changes he/she wants to review and in which cases.

Using fig. 1 as an example again, the *Manufacturer* might choose a high trust level for the *Warehouse* and *Logistics* as both are departments of the same company. The *Retailer* is under contract and a long-known business partner, therefore the *Manufacturer* would choose the third level to assure that he/she gets notified if the *Retailer* changes anything before the any own activity is executed. The *Subcontractor* is a new partner the *Manufacturer* has no experience with. So he/she sets the trust level to 1, meaning each change will be presented for review. In this way, each participant can set their levels according to their own needs.

4 Evaluation

In order to prove the concept, a prototype of the described negotiation platform was implemented. The prototype is based on a private Ethereum blockchain using Solidity smart contracts.

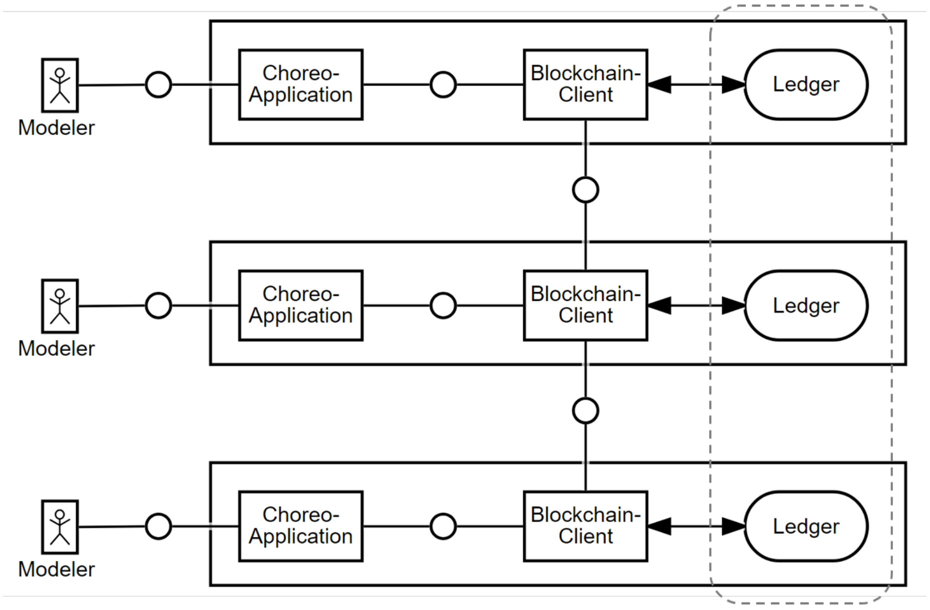

Fig. 3. Architectural overview

– Describe architecture and technologies
– Add screenshot and maybe link to screen cast
– Technologies used

4.1 Architecture

- Each participant has local installation of front end and back end
- front end is web interface to model and review
- back end also includes an Ethereum client with ledger, so each modeler is a node in the blockchain network too

4.2 Smart contracts

- Smart contracts written in Solidity
- Support flow described in approach
- Additional security in contract by guards
- One contract per diagram

4.3 Verifiers as a trusted party

As long as the smart contract is not able to determine the list of needed reviewers for a change on its own, adding the reviewers using a contract function, exposes a security risk: A participant could change the logic of the program locally, leading to an incomplete or even incorrect list of reviewers for his/her own proposal. This would make it possible for him/her, to manipulate the model without reviews and approvals of any other party.

In order to minimize this risk, we introduced another role in the negotiation process, the verifiers. As soon as a collaborator proposed a change and the list of reviewers was set, the smart contract determines a couple of verifiers that need to approve the list. Of course, this can not be a manual task. Since every collaborator deploys the platform locally, including the always running back end, this task can be performed automatically. Once the contract assigned a collaborator to be a verifier, his/her back end will compute the list of reviewers on its own, based on the proposed change and give feedback to the contract whether it came to the same result or not. Only if all verifiers agreed that the list is correct, the proposal will be opened for the reviewers, otherwise it will be rejected.

5 Discussion

- Conclusion
- Drawbacks of implementation:
- - ToDo: Store binary tree on blockchain
- - ToDo: Analyse change (as binary tree) on blockchain, regarding reviewers and verifiers
- Future work including:
- - Implementation of additional modeling elements
- - Embedding the communication relations of participants into trust levels
- - Future future: Versioning, branching

References

1. OMG: Business Process Model and Notation (BPMN), Version 2.0 (January 2011)