TRANSFORMATION OF THE BPMN BUSINESS PROCESS MODEL INTO SMART CONTRACTS FOR THE HYPERLEDGER FABRIC ENVIRONMENT

JANKO HRIBERŠEK

University of Maribor, Faculty of organizational sciences, Kranj, Slovenia; e-mail: jhribersek@gmail.com

Abstract Transformation of BPMN business process model will be a very important topic in the future of the Hyperledger Fabric blockchain environment. Machine transformation can increase the quality of transformation and thus reduce errors. The research paper first describes BPMN and the Hyperledger Fabric environment, what smart contracts are and why they are so important. In the second part, the raw transformation model is described, where the inputs for the transformations are BPMN and the metadata file, and the result is a smart contract written in Java that can be imported into the Hyperledger Fabric environment.

Keywords: BPMN, transformation, blockchain, hyperledger

University of Maribor Press

DOI https://doi.org/10.18690/978-961-286-485-9.53
ISBN 978-961-286-485-9

1 Introduction

Blockchain technology is revolutionizing the creation of scalable information systems and diverse applications by incorporating increasingly popular artificial intelligence, cloud computing, and large databases (Lu, 2019). The possibilities of using blockchain technology are being explored by various industries, so it is expected to spread to all sectors of industry. The proposal addresses the problem of machine conversion of a business process model into an executable language with Hyperledger Fabric smart contracts for blocks with permissions, where the model is written in the standard notation for BPMN 2.0 modelling. Solving this problem would significantly reduce the time required to convert a business process into a smart contract. This would reduce conversion costs, increase the security of business processes, and reduce the number of errors when converting to executable code. The adoption of blockchain technology would increase the traceability of the process. There would be no need to create an additional audit trail because records in the blockchain are never deleted. They are only added, and, in this way, it is always possible to revisit all stored values in the blockchain. There are few entries in the literature on this transformation (López-Pintado et al., 2018), and a similar experiment from 2019 (Lopez-Pintado et al., 2019) is related to BPMN and the Ethereum environment. Therefore, in the following problem, we analyzed the properties of BPMN and the blockchain environment with a focus on Hyperledger Fabric.

2 Problem definition

2.1 Business Process Modelling and Notation (BPMN)

The Object Management Group (OMG, 2013) has developed a standard Business Process Model and Notation (BPMN). The main goal of BPMN is to provide a record that is easily understood by all business users, from the business analysts who create the initial process designs to the technical developers responsible for implementing the technology that will implement those processes to the business people who will manage and monitor those processes. BPMN creates a standardized bridge for the gap between business process design and implementation. Another goal of BPMN is to ensure that eXtensible Markup Language (XML) languages designed for business process execution, such as WSBPEL (Web Services Business

Process Execution Language), can be visualized with a business-oriented data set. BPMN provides an easy way to share process information with other companies, users, process contractors, customers, and suppliers.

BPMN is widely used to graphically represent business process artifacts such as start, end, flow, activity, event, and transition. The model, written in BPMN syntax, formally represents a graph with nodes and links. The BPMN syntax also allows the definition of conditions for the transition from one node to another. Any markup language can be used to write it. The Object Management Group used the scalable XML markup language (OMG, 2013) in its development. The markup language itself has no execution capability, so BPMN is used as the top, contextual layer for automated mapping into an executable language. The middle layer of mapping is called Business Process Execution Language (BPEL) and at the time of its creation supported Service Oriented Architecture (SOA) or even more explicitly Web-based Services (WEB service) via the Web Service Description Language (WSDL). The lowest layer is an executable language that can communicate with a WEB service usually this is bytecode translated into Java. The process of orchestrating a business process—that is, transforming BPMN to executable (Java) code is only partially automated, time-consuming, and the risk of errors is high.

2.2 Blockchain technology

In the near future, blockchain technology will become a powerful tool for use in Industry 4.0 as it integrates and brings together architecture, technology, devices, and other related things to deliver high-quality products and services (Lu, 2019). Blockchain will also be important in the upcoming Industry 5.0. It is a new production model based on the interaction between people and machines. It emphasizes the standard of living, creativity and high quality of customized products. (Rada, 2018)

Blockchain technology became interesting with the release of the first cryptocurrency, Bitcoin. The electronic payment system is based on cryptological evidence rather than trust and allows two willing parties to transact directly without the involvement of a trusted third party (Nakamoto, 2008). From the cryptocurrency environment, blockchain technology has leapt to a higher level with the advent of the Ethereum platform, which allows for decentralized application replication. The novelty of the Ethereum platform was the smart contracts implemented in the Solidity and Vyper languages. The further development of blockchain technology is moving towards private blockchain environments, where users are authenticated when they integrate into the blockchain environment (private environment or permissioned environment).

The immutability and transparency of blockchain technology reduces human error and the need for manual intervention in database conflict situations. Blockchain can help streamline business processes by eliminating the duplication of data management activities (Tuan et al., 2018). Despite its popularity, blockchain technology still faces fundamental problems in managing transactions that are somewhat like those of traditional databases. Sharma et al. (2019) cite the Hyperledger Fabric blockchain platform as an example where parallel transaction processing can be compared to concurrency control mechanisms in traditional databases (Sharma et al., 2019). In a permissioned environment, a smart contract plays an important role. The Hyperledger Fabric platform is one of the representatives of the latest generation of blockchain environments, which aims to upgrade cross-industry technologies. In this cross-industry environment, smart contracts supported by Hyperledger Fabric play an important role.

In addition, the Hyperledger Fabric platform contains many advanced functionalities. It is possible to run distributed applications written in general standard programming languages without depending on the internal cryptocurrency. In previous blockchain platforms, smart contracts were written in specific languages. The flexibility is enabled in Hyperledger Fabric with the blockchain design itself and does not consume resources or reduce hardware performance (Androulaki et al., 2018).

Information solutions based on blockchain technology have been introduced in quite different sectors over the last five years. Solutions have been released in electricity marketing (Knirsch et al., 2019; Silva et al., 2019), finance (Duong-Trung et al., 2019; G. S. Group, 2016; Kabra et al., 2020; Nguyen et al., 2020; Wang et al., 2019), computing (Elghaish et al., 2020), law (Truong et al., 2019), food production and processing (Kumar et al., 2020), manufacturing (Sund et al., 2020), transportation (Naerland et al., 2017), and healthcare (Niu et al., 2020; Tanwar et al., 2020).

2.3 Smart contracts in the Hyperledger Fabric environment

A smart contract is a business logic that operates on a chain of blocks. They can be as simple as updating data or as complex as implementing built-in contract terms. There are two different types of smart contracts (*Hyperledger Architecture, Volume II, Smart Contacts*, n.d.):

- With built-in smart contracts the business logic of network certifiers is established before the network goes live.
- On-chain smart contracts introduce the business logic as a transaction that is added to the blockchain and executed each time the transaction is invoked. With on-chain smart contracts, the execution code of the business logic becomes part of the ledger.

The implementation of smart contracts in the Hyperledger Fabric environment is divided into three segments:

- The input contains the contract code, the transaction request, the possible dependency of the transaction, and the current state of the general ledger.
- The contract interpreter contains the current state of the general ledger and the program code of the smart contract.
- Outputs are generated only if the request was correct and confirmed. The output contains the new status and the side effects of the executed smart contract.

The smart contract layer is responsible for processing transaction requests and determining whether transactions are valid according to the specified business logic (*Hyperledger Architecture, Volume II, Smart Contacts*, n.d.). It validates each request by ensuring compliance with the policy and contract for a given transaction. Invalid requests are rejected and may be excluded from the block depending on the framework.

Transactions are considered either not yet begun or not yet completed. The transaction cannot be completed in parts. This ensures the integrity of transactions. Smart contracts can determine the dependencies between multiple transactions that need to be executed individually. These dependencies can be implicit or explicit.

With implicit reference, it is usually impossible to determine the order of transactions. Bitcoin solves this by repeatedly attempting to use the transaction, which results in transactions being executed one at a time until their preconditions are met. Such behaviour requires a transaction transfer component (pool, mempool) to take care of uselessly expiring transactions. With an explicit reference, the user specifies the transaction identifier.

2.1.1 Communication of a smart contract with other architectural layers

The smart contract layer works closely with the consensus layer. Specifically, the smart contract layer receives a proposal from the consent layer. This proposal specifies which contract to execute, the details of the transaction, including the identity and credentials of the entity requesting the execution of the contract, and any transaction dependencies.

The smart contract layer uses the current general ledger balance and the entry from the consent layer to confirm the transaction.

During transaction processing, the smart contract layer uses the identity service layer to authenticate and authorize the entity requesting the smart contract execution. This ensures two things: The entity is known on the blockchain network and the entity has sufficient access to execute the smart contract. Identity can be provided through a variety of methods: simple identities, ledger-managed identities and credentials, anonymous credentials, or managed identity services from a third-party certificate authority.

After processing the transaction, the smart contract layer returns the transaction acceptance status (transaction accepted or rejected). If the transaction is accepted, the smart-contract layer also returns the regularity certificate, delta state, and any optional ordering instructions required for transaction dependency compliance. Delta states include sets of changes and any side effects that should occur once the partners have successfully completed the transaction.

2.1.2 Integrity and availability of smart contracts

To ensure the integrity and availability of the blockchain network and the smart contract layer, enterprise blockchain networks must control access to certain resources. Since smart contracts are programs, they are exposed to malicious attacks, coding errors, and poor design. Disrupting the execution of program code in any of these areas can compromise the integrity or availability of the blockchain system.

Hyperledger recommends the following four security precautions for use with a smart contract layer to ensure integrity and availability:

- Denial of Service Protection
- Sandboxing
- Resource Management / Flow Control and Application Lifecycle Management (ALM)

In the design phase of this predisposition, we found two recent studies on the successful transformation of BPMN to Ethereum (Lopez-Pintado et al., 2019; López-Pintado et al., 2018), and none on the transformation of Hyperledger Fabric, which is a more technology-aware technology.

3 Methodology

To conduct the research, we will use:

- Study the relevant literature on BPMN and Hyperledger Fabric.
- Software engineering, which will include problem definition, programming and testing using the white box method.
- Verification and validation of the developed solution on selected cases of business processes. Verification (checking conformance to specification) will be performed after each activity that results in a test artifact and validation will be performed with key stakeholders of a given business process.

In the context of software engineering, a core and language for software engineering methods called Essence is used, which was developed by the Object Management Group (Object Management Group, 2018). It is the result of the SEMAT initiative (Software Engineering Methods and Techniques), which states on its website (*Http://Semat.Org*, n.d.) that Essence can be used by researchers to define a problem that needs to be understood and explored as part of an effort to develop a general theory of software engineering (cit. " Researchers can use Essence as a definition of the problem they want to understand and explore in their efforts to develop a General Theory of Software Engineering."). The current version of the Essence core and language is 1.2, with beginnings dating back to 2012 (Jacobson et al., 2012). Ivar Jacobson et. al. (Jacobson et al., 2019) state that software complexity is not the only reason for the "software crisis". They believe that developing solutions is not only about programming, but also about planning activities, managing a group of stakeholders, and effective communication and cooperation (p. 18). Ian Sommerville in the preface of the publication talks about Essence as a metamethod because the universality of the concepts involved allows it to be used in a broader range of domains than current methods, and Grady Booch says that he and his friends Ivar Jacobson and Jim Rumbaugh have tried to help developers with UML (Universal Modelling Language), in which some things are set right, others wrong, and that Essence is considered a set of basic software engineering abstractions. Jacobson et al. (2019) freely admit that software engineering is still in the process of developing a suitable theory that encompasses both descriptive and predictive aspects. Essence is seen as an important descriptive theory, while Tarpit (Johnson & Ekstedt, 2016) is ascribed the role of predictive theory. The core of Essence otherwise includes:

- *Alphas* - descriptions of things we manage, develop, and use in the process of development, maintenance, and support. Alphas can have subalphas,
- *Activity Spaces* - representations of important things that need to be done in the process of developing, maintaining, and supporting software solutions; and
- *Competencies* - a representation of the key skills required for all activities.

In the current state of research (dissertations) are:

- *Alpha* - business process models in BPMN, transformation rules, smart contracts in the chosen programming language (probably Java and/or Javascript), a set of metadata about the business processes under study,
- *Activity spaces* - detailed acquaintance with Hyperledger Fabric, contexts of business process models, variants of developed program code and white-box testing, and
- *Competences* - explicit and implicit knowledge of programming languages BPMN, Hyperledger Fabric, Java and Javascript.

The Essence language specification includes:

- *Headings* - formal names of language elements,
- *Descriptions* - an informal description of a language element,
- *Generalizations* - a structure of classes and parent classes,
- *Attributes* - a list of attributes with their data types,
- *Associations* - a list of links that the language element has,
- *Invariants* - informal verbal descriptions of correctly formed rules and Object Constraint Language (OCL) terms,
- *Additional Operations* - descriptions of additional operations for correctly formed rules, and
- *Semantics* - a detailed description of the elements in natural language.

Language specifications with Essence cannot be defined well enough at the current stage of research, but they will be developed at a later stage.

4 Expected results

The primary research question of the dissertation proposal is:

How can the business process model described in BPMN with certain metadata be automatically and formally converted into the execution code of smart contracts in the Hyperledger Fabric blockchain environment?

In line with the research question, the objectives are:

- investigate BPMN and Hyperledger Fabric.
- in the case of a business process, define the steps and procedures for transforming BPMN into smart contracts.
- test the sequence of steps and procedures for the transformation of several other standard business processes.
- identify the minimum set of metadata relevant for the transformation.
- develop software that formally correctly performs the transformation from BPMN to Hyperledger Fabric smart contracts with permitted access.

The achieved goals of the research will enable the understanding of the process of language conversion for data specification into a Turing universal machine. The developed approach and application will be useful in:

- the basic research of business informatics
- practice for business management, specifically for enhancing business performance.

The transformation of the business process model from BPMN to a smart contract will be presented and tested in detail. A minimal set of metadata is identified to define the details in the Hyperledger Fabric environment.

5 Future development

At this point, development is still in the first stage. Further development will include the following development steps:

- development of a software module for the identification of individual BPMN building blocks (event, activity, gateway, sequence flow)
- development of a transformation library for individual basic BPMN building blocks.

- development of a solution that prepares a transcription of the BPMN model from the XML format into an intermediate format arranged in the order of implementation of the individual activities captured in the BPMN.
- a transformation program that will create software in Java from a list of sequential activities and using a metadata file.

References

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., … Yellick, J. (2018). Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. 1–15. https://doi.org/10.1145/3190508.3190538

Duong-Trung, N., Ha, X. S., Phan, T. T., Trieu, P. N., Nguyen, Q. N., Pham, D., Huynh, T. T., & Le, H. T. (2019). Multi-sessions mechanism for decentralized cash on delivery system. International Journal of Advanced Computer Science and Applications, 10(9), 553–560. https://doi.org/10.14569/ijacsa.2019.0100973

Elghaish, F., Abrishami, S., & Hosseini, M. R. (2020). Integrated project delivery with blockchain: An automated financial system. Automation in Construction, 114(November 2019), 103182. https://doi.org/10.1016/j.autcon.2020.103182

G. S. Group. (2016). Blockchain: The New Technology of Trust. https://www.goldmansachs.com/insights/pages/blockchain/

http://semat.org. (n.d.). Retrieved September 10, 2020, from http://semat.org

Hyperledger Architecture, Volume II, Smart Contacts. (n.d.). Retrieved January 6, 2019, from https://www.hyperledger.org/wp-content/uploads/2017/08/Hyperledger_Arch_WG_Paper_1_Consensus.pdf

Jacobson, I., Lawson, H. "Bud," Ng, P.-W., McMahon, P. E., & Goedicke, M. (2019). The Essentials of Modern Software Engineering: Free the Practices from the Method Prisons! In The Essentials of Modern Software Engineering: Free the Practices from the Method Prisons! Association for Computing Machinery. https://doi.org/10.1145/3277669

Jacobson, I., Ng, P. W., McMahon, P. E., Spence, I., & Lidman, S. (2012). The essence of software engineering: The SEMAT kernel. Queue, 10(10), 40–51. https://doi.org/10.1145/2381996.2389616

Johnson, P., & Ekstedt, M. (2016). The Tarpit - A general theory of software engineering. Information and Software Technology, 70, 181–203. https://doi.org/10.1016/j.infsof.2015.06.001

Kabra, N., Bhattacharya, P., Tanwar, S., & Tyagi, S. (2020). MudraChain: Blockchain-based framework for automated cheque clearance in financial institutions. Future Generation Computer Systems, 102, 574–587. https://doi.org/10.1016/j.future.2019.08.035

Knirsch, F., Unterweger, A., & Engel, D. (2019). Implementing a blockchain from scratch: why, how, and what we learned. Eurasip Journal on Information Security, 2019(1). https://doi.org/10.1186/s13635-019-0085-3

Kumar, A., Liu, R., & Shan, Z. (2020). Is Blockchain a Silver Bullet for Supply Chain Management? Technical Challenges and Research Opportunities. Decision Sciences, 51(1), 8–37. https://doi.org/10.1111/deci.12396

Lopez-Pintado, O., Dumas, M., Garcia-Banuelos, L., & Weber, I. (2019). Interpreted execution of business process models on blockchain. Proceedings - 2019 IEEE 23rd International

Enterprise Distributed Object Computing Conference, EDOC 2019, 206–215. https://doi.org/10.1109/EDOC.2019.00033

López-Pintado, O., García-Bañuelos, L., Dumas, M., Weber, I., & Ponomarev, A. (2018). Caterpillar: A business process execution engine on the Ethereum blockchain. Software - Practice and Experience, 00, 1–45. https://doi.org/10.1002/spe.2702

Lu, Y. (2019). The blockchain: State-of-the-art and research challenges. Journal of Industrial Information Integration, 15(January), 80–90. https://doi.org/10.1016/j.jii.2019.04.002

Naerland, K., Müller-Bloch, C., Beck, R., & Palmund, S. (2017). Blockchain to Rule the Waves - Nascent Design Principles for Reducing Risk and Uncertainty in Decentralized Environments. Icis, 1–16. https://www.researchgate.net/profile/Christoph_Mueller Bloch/publication/319990622_Blockchain_to_Rule_the_Waves_-_Nascent_Design_Principles_for_Reducing_Risk_and_Uncertainty_in_Decentralized_Environments/links/59c8df97aca272c71bcdc61f/Blockchain-to-Rule-t

Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Journal for General Philosophy of Science, 39(1), 53–67. https://doi.org/10.1007/s10838-008-9062-0

Nguyen, B. M., Dao, T. C., & Do, B. L. (2020). Towards a blockchain-based certificate authentication system in Vietnam. PeerJ Computer Science, 2020(3). https://doi.org/10.7717/peerj-cs.266

Niu, S., Chen, L., Wang, J., & Yu, F. (2020). Electronic Health Record Sharing Scheme with Searchable Attribute-Based Encryption on Blockchain. IEEE Access, 8, 7195–7204. https://doi.org/10.1109/ACCESS.2019.2959044

Object Management Group. (2018). Kernel and Language for Software Engineering Methods (Essence). 2007 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis, Versión 1.2, 300. https://www.omg.org/spec/Essence/1.2

OMG. (2013). Object Management Group, Business process modeling notation (BPMN) version 2.0.2. Object Management Group, December, 134. https://doi.org/10.1007/978-3-642-33155-8

Rada, M. (2018). INDUSTRY 5.0 definition – Michael Rada – Medium. https://medium.com/@michael.rada/industry-5-0-definition-6a2f9922dc48

Sharma, A., Agrawal, D., Schuhknecht, F. M., & Dittrich, J. (2019). Blurring the lines between blockchains and database systems: The case of hyperledger fabric. Proceedings of the ACM SIGMOD International Conference on Management of Data, 105–122. https://doi.org/10.1145/3299869.3319883

Silva, F. C., Ahmed, M. A., Martínez, J. M., & Kim, Y. C. (2019). Design and implementation of a blockchain-based energy trading platform for electric vehicles in smart campus parking lots. Energies, 12(24). https://doi.org/10.3390/en12244814

Sund, T., Lööf, C., Nadjm-Tehrani, S., & Asplund, M. (2020). Blockchain-based event processing in supply chains—A case study at IKEA. Robotics and Computer-Integrated Manufacturing, 65(March), 101971. https://doi.org/10.1016/j.rcim.2020.101971

Tanwar, S., Parekh, K., & Evans, R. (2020). Blockchain-based electronic healthcare record system for healthcare 4.0 applications. Journal of Information Security and Applications, 50. https://doi.org/10.1016/j.jisa.2019.102407

Truong, N. B., Sun, K., Member, S., Lee, G. M., & Member, S. (2019). GDPR-Compliant Personal Data Management : A Blockchain-Based Solution. 15(March), 1–13.

Tuan, T., Dinh, A., Liu, R., Zhang, M., Chen, G., Dinh, T. T. A., Liu, R., Zhang, M., Chen, G., Ooi, B. C., & Wang, J. (2018). Untangling blockchain: A data processing view of blockchain systems. IEEE Transactions on Knowledge and Data Engineering, 30(7), 1366–1385. https://doi.org/10.1109/TKDE.2017.2781227

Wang, H., Guo, C., & Cheng, S. (2019). LoC — A new financial loan management system based on smart contracts. Future Generation Computer Systems, 100, 648–655. https://doi.org/10.1016/j.future.2019.05.040