Tiako P.F. (ed) Automated Systems, Data, and Sustainable Computing. Chronicle of Computing. OkIP.
 https://doi.org/10.55432/978-1-6692-0001-7_9

Blockchain Interoperability Method for the Cross-organizational Transactions: A Meta-Meta-Model

Ebtehal Y. Nassar[a*], Iman M. A. Helal[a], Sherif Mazen[a]

[a]*Faculty of Computers and Artificial Intelligence, Cairo University, Giza, 12613, Egypt*

ABSTRACT

Nowadays, blockchain systems are attracting attention in both academic and industry fields. Their main advantages are to their provided capabilities in security, and immutability. However, these systems are still immature growth, as there is a lack of templates for blockchain development. Interoperability can be considered one of the challenges while developing new blockchain systems, due to the heterogeneity of the blockchain systems infrastructure, consensus protocol, privacy level, etc. Although many organizations aim to utilize blockchain systems, they suffer from the limitations of running cross-organizational transactions over different blockchain systems. The current cross-organizational solutions depend on how to make a connection between the different organizations' applications to reach choreography. In this paper, we will address the current interoperability challenges in blockchain systems and their impact on the running of cross-organizational transactions. Moreover, we will illustrate the usage of Model-Driven Engineering (MDE) to model the different aspects of blockchain systems. Finally, we will present an MDE-based solution to reach organizational interoperability as a basis for modeling hybrid applications that can process cross-organizational transactions.

Keywords: Blockchain, Interoperability, Model-Driven Engineering, Meta-Meta-Model, Cross-Organizational transaction, Smart Contracts

I. INTRODUCTION

Blockchain systems offer new capabilities like immutability and secure transactions, that trigger most of the organizations to join blockchain networks or implement their own blockchain. Although, the implementation of blockchain networks became trendy, the completely shift for blockchain systems is not a one fits all solution (Zheng, Z., Xie, S., Dai, H.N., Chen, X. and Wang, H., 2018). Because of the diversities that blockchain systems have in implementation like consensus algorithms, distributed ledger and block structures, smart contract development, … etc. Also, these systems differ from the regular systems (object-oriented applications). This leads to the interoperability problem (Scheid, E.J., Hegnauer, T., Rodrigues, B. and Stiller, B., 2019), because of the limitations of connecting these different blockchain systems together, and correspondingly connecting them with non-blockchain systems. Developing blockchain applications is far from easy, and mistakes may not be fixable.

Many business processes need collaboration between different organizations to be completed. These processes are the basis for systems like supply chain management systems, healthcare systems, governmental systems, etc. The collaboration between the previously mentioned systems can be reached by applying cross-organizational transactions. Each of these systems have different systems' structures; therefore, the interoperability is a challenge to apply cross-organizational transaction (Xu, X., Bandara, H.D., Lu, Q., Weber, I., Bass, L. and Zhu, L., 2021).

In this research, we show how interoperability is a challenge in blockchain, we describe the different states where cross-organizational transactions are vital, comparing the main categories of applying inter-organizational transactions, and illustrate the usage of MDE techniques and methods in blockchain systems (Lu, Q., Binh Tran, A., Weber, I., O'Connor, H., Rimba, P., Xu, X., Staples, M., Zhu, L. and Jeffery, R., 2021).

Verifying the correctness of the model can be easier than the verifying of raw code. Using specific tools can ensure that the deployed code has not been changed after being derived from the model. Nevertheless, the code generation tool needs to be correct. Thus, we investigate the effect of applying MDE to solve one of the interoperability challenges for the cross-organizational transaction.

Our research problem is focused on the blockchain inability to handle the following:

- Integrate blockchain data with the organization's existing systems.
- Ensure a unified model to exchange data between the organization applications and the different types of blockchains (Hsain, Y.A., Laaz, N. and Mbarki, S., 2021).

The paper is structured as follows: section II defines the main concepts and background knowledge regarding interoperability problem in the blockchain field, the current challenges, the suggested solutions, the different challenges of cross-organizational transactions and the usage of MDE in modelling blockchain different aspects, while section III introduces our suggested hybrid application model. Section IV materializes related work. Finally, concluding our work and future work in section V.

II. BACKGROUND

In this section, we clarify the interoperability problem from blockchain perspective (sub-section A), and how cross-organizational transactions can present many challenges to design and model smart contracts (sub-section B). Finally, we present the current MDE techniques (sub-section C).

* Ebtehal.yahia@fci-cu.edu.eg

A. Blockchain Interoperability

The *interoperability* can be defined as follows: It is the capability to make autonomous systems communicate to exchange information and services, despite having variations in different parameters (programming language, execution platform, interface, etc.) (Lohachab, A., Garg, S., Kang, B., Amin, M.B., Lee, J., Chen, S. and Xu, X., 2021).

Piking new, innovative blockchain systems enables developers and users to gain the benefits of state-of-the-art technology. However, the lack of user experience, at the start, makes it a risk to use novel blockchain, due to the possibility of security breaches (Pang, Y., 2020). On the other side, using mature, reliable blockchains reduces the risk of losses and failures, because those blockchains pass by different analysis stages, but they do not have the features of the novel blockchains.

In some use cases, a different blockchain may be more suitable because of requirements and/or circumstances change. In addition, the blockchain system may become obsolete, attacked, or out of service, in order to handle such cases a user should be able to save his assets by transferring from a blockchain to another (Hewett, N., Lehmacher, W. and Wang, Y., 2019).

Also, it is a challenge to organize transactions from different blockchains to enable cross-chain distributed applications, if different blockchains have various properties . Especially if we try to revert a transaction that depends on another given different transaction finalities from different blockchains (Langer, A.M., 2020).

The blockchain interoperability methods were described from a granularity perspective in the following categories: (Isolated interoperability, network interoperability, structural interoperability, semantic interoperability, specification interoperability, platform interoperability, and organizational interoperability). Our focus is on the organizational interoperability as some organizations support blockchain as their primary technology because it is more secure than other legacy systems. But they may use various blockchain networks based on their needs. Hence, they must establish a communication way to exchange specific information.

B. Cross-organizational Transactions

Several research proposals have demonstrated the feasibility of designing blockchain-based collaborative business processes using a high-level notation, such as the Business Process Model and Notation (BPMN), and thereon automatically generating the code artifacts required to execute these processes on a blockchain platform (Xu, X., Bandara, H.D., Lu, Q., Weber, I., Bass, L. and Zhu, L., 2021).

lack of appropriate inter-blockchain communication limits the adoption of blockchain. Blockchain technology could become a reasonable solution for most systems if it can scale and communicate with other systems. That makes the need to have a mechanism that would connect with multiple entities' blockchain systems without any intermediary or broker. At the same time, it's required to preserve the property of trust and integrity for each blockchain (Pillai, B., Biswas, K. and Muthukkumarasamy, V., 2020).

The usage of blockchain in cross-organizational transactions is mainly for reaching choreography between different organizations. The applying of blockchain in such a system is to enact choreographies in a trust-less environment (Lichtenstein, T., Siegert, S., Nikaj, A. and Weske, M., 2020). The presented solutions utilize the model-driven concepts to use the BPMN choreography model and generate smart contracts code, for example ChorChain system as presented in (Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B. and Tiezzi, F., 2020). Another perspective is to use blockchain as a software connector as in (De Sousa, V.A. and Corentin, B., 2019) to help organizations integrating the IT systems they use to support business processes. Its main limitation is the lack of integrated methodology that combine a consistent set of models to design and implement software connectors relying on blockchain to support the integration of IT systems used for cross-organizational BPs.

One of the proposed systems for solving these problems is the Ethereum-based process execution framework for cross-organizational process collaborations. Smart contracts are used as a mechanism to enforce a trusted and immutable process flow making the need for a TTP (Trusted Third Party) obsolete (Heine M, Poustcchi K, Krasnova H, Klinger P, Bodendorf F.2020).

The previously presented solutions are suitable for handling the collaboration between different organizations using a blockchain system. However, to the best of our knowledge, there are no trials to model a design for the organization application itself, which is called organizational interoperability (Section II). It is important to model organizations' systems that can be used to join blockchain networks.

C. Model-Driven Engineering Techniques

Model-driven engineering (MDE) is a software engineering methodology using models with various views and levels of abstraction for different purposes during software development. Models with a low level of abstraction can directly generate software production code, while those with a high level of abstraction can provide guidance and even support system analysis before implementation. MDE proposes a high level of abstraction representation to address heterogeneity and system complexity (Hsain, Y.A., Laaz, N. and Mbarki, S., 2021).

MDE is very useful for giving standards to software building. It also presents automated verification and analysis tools, improved development productivity, and guaranteeing compliance through correct design (Wöhrer, M. and Zdun, U., 2020).

MDE is considered one of the most used techniques for solving software modelling problems. Hence, it's heavily used for modelling and designing blockchain systems. We can categorize the MDE usage in the blockchain field as follows: model the blockchain network, model the smart contracts, model the distributed ledger, and model the blockchain applications (Mao, D., Wang, F., Wang, Y. and Hao, Z., 2019).

1) Model Blockchain Network: In (Abbas, M., Rashid, M., Azam, F., Rasheed, Y., Anwar, M.W. and Humdani, M., 2021), the authors introduce a novel and efficient framework that is

based on model-driven architecture. Particularly, a meta-model (Ecore[1] Model) is defined that contains the concepts of Blockchain technology. As a part of tool support, a tree editor (developed using Eclipse Modeling Framework[2]) and a Sirius-based graphical modeling tool with a drag-drop palette have been provided to allow modeling and visualization of simple and complex blockchain-based scenarios for security labs in a very user-friendly manner. A Model to Text (M2T) transformation code has also been written using Acceleo[3] language that transforms the modeled scenarios into java code for blockchain applications.

2) *Model Smart Contracts:* Due to the conceptual discrepancy between contractual clauses and corresponding code, it is hard for domain stakeholders to easily understand contracts, and for developers to write code efficiently without errors (Wang, S., Ouyang, L., Yuan, Y., Ni, X., Han, X. and Wang, F.Y., 2019.). The design of a domain-specific smart contract language is based on a higher level of abstraction that can be automatically transformed to an implementation. In (Wöhrer, M. and Zdun, U., 2020) a DSL (Domain Specific Language) was proposed, for generating code to Solidity language only, which is the Ethereum programming language. Due to complex of blockchain-based contract execution, the lack of programming abstractions, and the constant changes in the platform capabilities and security aspects, it became difficult to write smart contracts efficiently. The same authors presented smart contract design patterns and their automated application, using code generation and the use of a domain-specific language. But it is only compatible with Ethereum blockchain networks (Wohrer, M. and Zdun, U., 2020).

The authors of (De Sousa, V.A., Burnay, C. and Snoeck, M., 2020) proposed B-MERODE as an MDE approach to generate smart contracts supporting cross-organizational collaborations. Its target is to develop smart contracts to facilitate the development and improvement of cross-organizational business processes. It is considered a novel approach relying on MDE and artifact-centric business processes to generate smart contracts supporting cross-organizational collaborations.

3) *Model Distributed Ledgers:* There is a need to provide modeling support for the deployment view of distributed ledger solutions. One of the earlier trials is presented in (Górski, T. and Bednarski, J., 2020), the authors present how to design transformation for generating deployment scripts for the R3 Corda Distributed Ledger Technology (DLT) framework with the ability to switch to another technology. Due to architectural differences between the distributed ledger platforms, this solution does not provide a transition to another platform without changes in the source code of the transformation. For the same purpose, in (GÓrski, T. and Bednarski, J., 2020) the authors use a description of the UML2Deployment transformation of the distributed ledger's deployment model into its deployment script. However, the transformation has been designed for the R3 Corda framework only.

4) *Model Blockchain Applications:* One of the tracks of applying MDE in the blockchain area is to model the blockchain applications. This is important for collaborative business processes. A typical class of applications uses blockchain for the management of cross- organizational business processes as well as assets. However, developing such applications without introducing vulnerabilities or bugs can add difficulty for developers, for example the deployed code is immutable and can be called by anyone with access to the network. MDE helps in reducing those risks, by combining proven code snippets as per the model specification, which is typically easier to understand than source code with all its implications (Lu, Q., Binh Tran, A., Weber, I., O'Connor, H., Rimba, P., Xu, X., Staples, M., Zhu, L. and Jeffery, R., 2021).

There are many trials of applying MDE techniques for modelling blockchain systems. The usage of domain-specific language and transformation to smart contract code can be considered as a gold-mine for blockchain developers. However, each proposed solution is tightly coupled to the blockchain platform it is designed for only, for example, Ethereum, Corda, etc.

III. THE PROPOSED HYBRID APPLICATION MODEL

In software engineering, models can be categorized as descriptive and prescriptive. The descriptive models are used for capturing knowledge, for example domain analysis, requirements, …etc. On the other hand, the prescriptive models are used as blueprints for system designs, and implementations. The main purpose of prescriptive models is planning and early errors discovery. In addition to, blueprints can be used for partially evaluating systems before realizing. It is one of the goals of MDE to shift the emphasis from informal, non-binding models to rigorous, binding models.

As mentioned in Section II, one of the interoperability methods is the organization interoperability between the different parts of the same application. In our work, we target this type of interoperability. In some business cases like supply chain management systems, the organization may need to join different networks and share data and processes with different architectures (Sánchez-Gómez, N., Torres-Valderrama, J., García-García, J.A., Gutiérrez, J.J. and Escalona, M.J., 2020). These networks may be blockchain networks with different platforms. Therefore, it's a challenge to make an organization application that can join different networks with fewer efforts and changes. This will save cost and development time (Hamdaqa, M., Metz, L.A.P. and Qasse, I., 2020). The current solutions for such challenges are to have a meta-model for a smart contract that can be translated into code. But these models work on modelling smart contracts only, without taking into consideration that the organization may need to use its legacy systems with the blockchains. Based on the multilevel modelling technique, we created a meta-model for the Class diagram , it consists of seven classes, we added more details by defining a class for generalization and another one for the association relationships. These meta-models represent the second tier of our architecture in Fig. 1 Meta-Meta-Model architecture for class diagram and smart contract. Both models of the second tier (M1) conform to the model of the first tier (M2). We developed the Meta-Models as Ecore models, that can help in automating code generation.

So, there is a need to have models that can apply for legacy systems, and new systems like blockchains altogether. The target of our work is to help the application designer to design a hybrid application model that can be translated later into classes, smart contracts, or both.

One of the mapping features of a model is that a model must have an origin. Therefore, we build a meta-meta-model for class diagram and smart contracts together to be an origin for both. We apply the multi-level meta modelling technique presented in (Lara, J.D., Guerra, E. and Cuadrado, J.S., 2014) to have different models' tiers. As mentioned in Fig. 1, the hierarchy of our design model is as follows: the higher tier will represent the Meta-Meta-Model of the class diagram and smart contract (M2 level). The second tier will be the class-diagram meta-model and the smart-contract meta-model (M1 level), the models in this tier conform to the M2 tier. The third tier will be the application model which may conform to the class-diagram model, smart-contract model, or both. The application model in M0 conforms to the meta-models in M1.

A. The Meta-Meta Model of Class Diagram and Smart Contract

The Meta-Meta-Model (Ecore Model) in Fig. 2 consists of the following classes: the main class is the ***Classifier*** class; this is the main class that the "class diagram" and "smart contract diagram" will inherit from. The ***Class*** here represents the main component of a class diagram, it is connected to the other main class diagram components (attribute, operation, parameter). The ***SmartContract*** class represents the modelling class of a smart contract, it is connected to the relationship through the classifier class, also connected to events through the relationship class. The SmartContract class is connected to the ***Element*** class which represents the elements that may exist inside a smart contract (asset, participant, transaction, ... etc.). The ***Event*** class is an event that may exist in smart contracts. The part specified with the class diagram is derived from the model represented in (Li, Y., Gu, P. and Zhang, C., 2014). The part related to the smart contract model is derived from the model represented in (Hamdaqa, M., Metz, L.A.P. and Qasse, I., 2020).

B. The Meta-Model of Class Diagram and Meta-Model of Smart Contract

Based on the multilevel modelling technique (Lara, J.D., Guerra, E. and Cuadrado, J.S., 2014), we created a meta-model for the Class diagram, cf. Fig. 3 it consists of seven classes, as

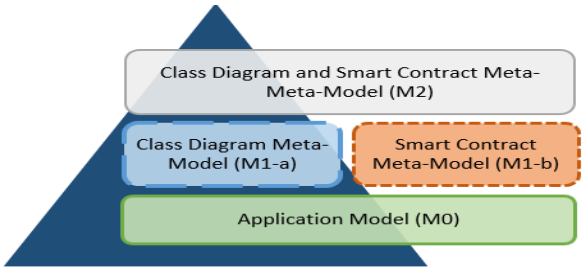

Fig 1 Meta-Meta-Model architecture for class diagram and smart contract

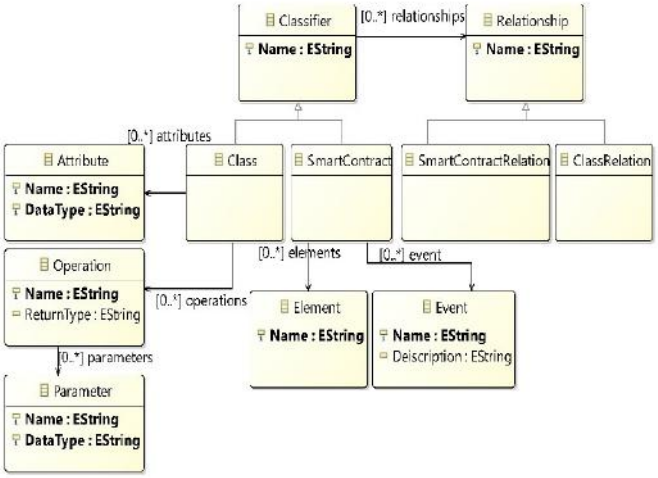

Fig 2 Meta-Meta-Model for Class Diagram and Smart Contract

we added more details by defining a class for generalization and another one for the association relationships. These meta-models represent the second tier of our architecture in Fig. 1. Both models of the second tier (M1) conform to the model of the first tier (M2). We developed the meta-models as Ecore models, that can help in automating code generation.

C. Supply Chain Management System Example

To illustrate our proposed hybrid application mode, we present a simple supply-chain-management model to apply our architecture. Its original class diagram is shown in Fig. 6. It consists of four classes (order, supplier, product, stock). We assumed that the supplier information should be recorded in a blockchain network using a smart contract, this is presented in Fig. 5, as a hybrid model of the application. Therefore, the supplier class will be implemented in a smart contract form, as presented in Fig. 5 block b. Which conforms to the smart contract meta-model. The other parts of the application conform to the class-diagram meta-model as shown in Fig. 5 block a. When it comes to the implementation, the application model in the third tier (M0) will have two interconnected parts; The first part (that conforms to the class diagram in Fig. 1 M1-a) will be translated to the specified programming language, the second part (that conforms to the smart contract in Fig. 1 M1-b) will be translated to the specified blockchain programming language.

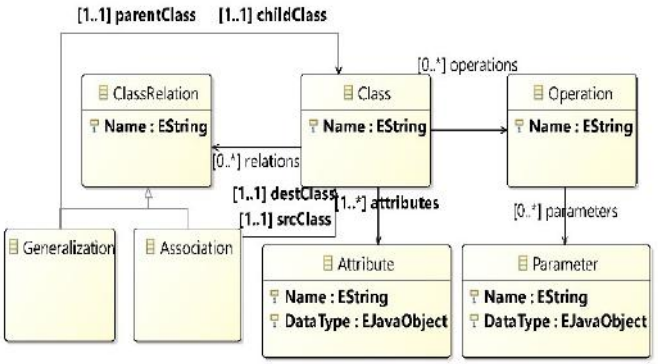

Fig 3 Class Diagram Meta-Model (M1-a)

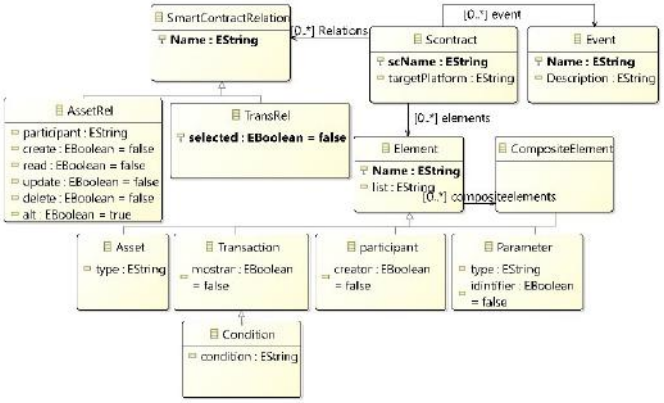

Fig 4 Smart Contract Meta-Model (M1-b)

IV. RELATED WORK

Most of the presented approaches in supporting cross-organizational processes view blockchain as a software connector helping organizations to integrate the IT systems, they use to support business processes. However, using blockchain as a connector proposed solutions did not mention how to design a hybrid application from the beginning to connect with various blockchain systems (Langer, A.M., 2020), which is our main emphasize in this paper. The authors of (De Sousa, V.A. and Corentin, B., 2019) suggested applying a model-driven engineering approach for the development of such connectors to design blockchain-based solutions that can be implemented on various blockchains using the same models. Based on this paper there is no methodology proposing an integrated set of models that can be connected to design and implement blockchain software connectors to support the integration of existing organizations' systems used for cross-organizational business processes. In (Hamdaqa, M., Metz, L.A.P. and Qasse, I., 2020) the authors presented a feature-oriented domain analysis approach to investigate the variations of most three famous blockchain platforms (IBM Hyperledger Composer, Azure Blockchain Workbench, and Ethereum). In addition to a well-structured model for smart contract. They also implemented a framework (iContractML) for transforming smart contract into code according to the target platform. The proposed framework models the smart contract based on the business requirements.

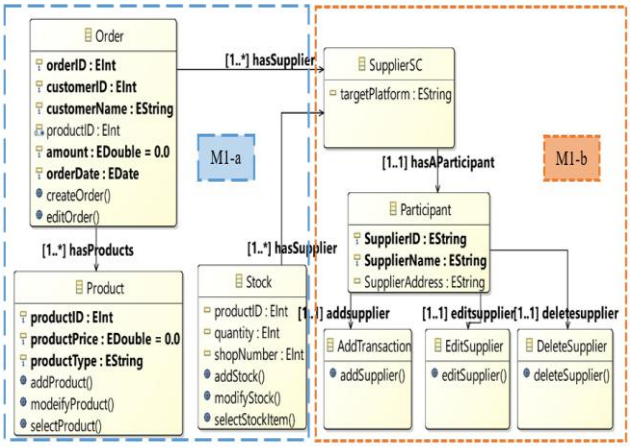

Fig 5 Supply Chain Management System Hybrid Application Model

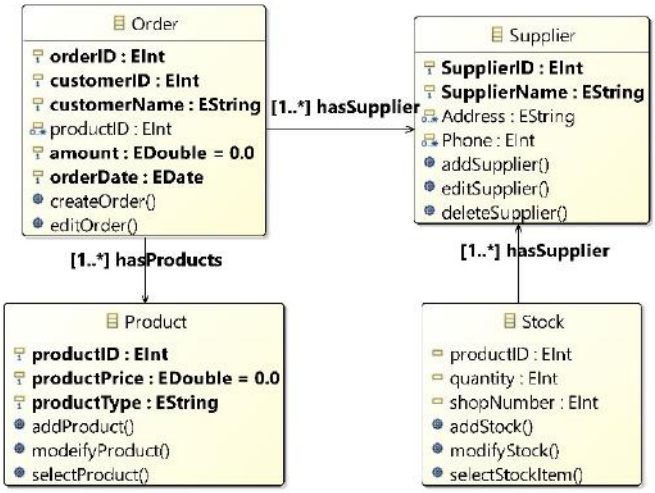

Fig 6 Supply Chain Management System Class Diagram

The difference between iContractML and our proposed solution is that iContractML generating model and code for one purpose application that can be implemented on one of the targeted blockchain platforms (Hyperledger, Azure, or Ethereum), but our proposed solution presents a model for hybrid application that can be composed of different blockchain smart contract codes at the same time. Based on the survey published in (Hsain, Y.A., Laaz, N. and Mbarki, S., 2021), which is a survey about the different MDE techniques for modelling Ethereum smart contracts, the authors stated that most research approaches did not mention the meta-model usage for modelling smart contract concepts. They also stated that the main category of modelling smart contracts interested by the behavioral aspect of smart contracts, modelling the business aspects by using UML state chart, and BPMN models. The second category focuses on modelling the static aspects using class diagram.

The third one designs the formal aspect of smart contract using: finite state machine model (Tolmach, P., Li, Y., Lin, S.W., Liu, Y. and Li, Z., 2021), OCL (Object Constraint Language), or ontologies (Syahputra, H. and Weigand, H., 2019). This study gives the importance of using meta-modelling for generating smart contract models.

Therefore, gives an advantage for our work as there are few research trials to use meta-models for modelling smart contracts.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented the different facets of interoperability in the blockchain area, in addition to the current solutions of cross-organizational transactions. We showed the benefits of Model-Driven Engineering and its different applications for modelling blockchain networks, smart contracts, distributed ledgers, and blockchain applications. Applying the multi-level modelling, we developed a meta-meta-model for modelling smart contracts and class diagrams to facilitate inheriting the same features as their parent. This led to modelling a hybrid application that implements and uses the classes and smart contracts. The developed model is the first step to have a general model for applying organizational

interoperability by having a general meta-meta-model. Building on this meta-meta-model we can create a transformation model to transform from class diagram to smart contract. This can enable developing a DSL to create smart contracts code automatically.

As future work, we will implement a UML profile [62] model that will enhance the usage of classes and smart contracts in the hybrid application. In addition to using this profile model as a basis for the transformation from classes into smart contracts.

REFERENCES

Zheng, Z., Xie, S., Dai, H. N., Chen, X., & Wang, H. (2018). Blockchain challenges and opportunities: A survey. International Journal of Web and Grid Services, 14(4), 352-375.

Scheid, E. J., Hegnauer, T., Rodrigues, B., & Stiller, B. (2019, October). Bifröst: a modular blockchain interoperability API. In 2019 IEEE 44th Conference on Local Computer Networks (LCN) (pp. 332-339). IEEE.

Xu, X., Bandara, H. D., Lu, Q., Weber, I., Bass, L., & Zhu, L. (2021, March). A decision model for choosing patterns in blockchain-based applications. In 2021 IEEE 18th International Conference on Software Architecture (ICSA) (pp. 47-57). IEEE.

Lu, Q., Binh Tran, A., Weber, I., O'Connor, H., Rimba, P., Xu, X., ... & Jeffery, R. (2021). Integrated model-driven engineering of blockchain applications for business processes and asset management. Software: Practice and Experience, 51(5), 1059-1079.

Hsain, Y. A., Laaz, N., & Mbarki, S. (2021). Ethereum's Smart Contracts Construction and Development using Model Driven Engineering Technologies: a Review. Procedia Computer Science, 184, 785-790.

Lohachab, A., Garg, S., Kang, B., Amin, M. B., Lee, J., Chen, S., & Xu, X. (2021). Towards interconnected blockchains: A comprehensive review of the role of interoperability among disparate blockchains. ACM Computing Surveys (CSUR), 54(7), 1-39.

Pang, Y. (2020). A new consensus protocol for blockchain interoperability architecture. IEEE Access, 8, 153719-153730.

Hewett, N., Lehmacher, W., & Wang, Y. (2019, April). Inclusive deployment of blockchain for supply chains. World Economic Forum.

Langer, A. M., Langer, & Wheeler. (2020). Analysis and Design of Next-Generation Software Architectures. New York: (pp. 149-164). Springer International Publishing.

Pillai, B., Biswas, K., & Muthukkumarasamy, V. (2020). Cross-chain interoperability among blockchain-based systems using transactions. The Knowledge Engineering Review, 35.

Lichtenstein, T., Siegert, S., Nikaj, A., & Weske, M. (2020, June). Data-Driven Process Choreography Execution on the Blockchain: A Focus on Blockchain Data Reusability. In International Conference on Business Information Systems (pp. 224-235). Springer, Cham.

Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B., & Tiezzi, F. (2020, March). Engineering trustable choreography-based systems using blockchain. In Proceedings of the 35th Annual ACM Symposium on Applied Computing (pp. 1470-1479).

De Sousa, V. A., & Corentin, B. (2019, May). Towards an integrated methodology for the development of blockchain-based solutions supporting cross-organizational processes. In 2019 13th International Conference on Research Challenges in Information Science (RCIS) (pp. 1-6). IEEE.

Klinger, P., & Bodendorf, F. (2020, March). Blockchain-based Cross-Organizational Execution Framework for Dynamic Integration of Process Collaborations. In Wirtschaftsinformatik (Zentrale Tracks) (pp. 1802-1817).

Wöhrer, M., & Zdun, U. (2020, May). Domain specific language for smart contract development. In 2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC) (pp. 1-9). IEEE.

Mao, D., Wang, F., Wang, Y., & Hao, Z. (2019). Visual and user-defined smart contract designing system based on automatic coding. Ieee Access, 7, 73131-73143.

Abbas, M., Rashid, M., Azam, F., Rasheed, Y., Anwar, M. W., & Humdani, M. (2021, April). A Model-Driven Framework for Security Labs using Blockchain Methodology. In 2021 IEEE International Systems Conference (SysCon) (pp. 1-7). IEEE.

Wang, S., Ouyang, L., Yuan, Y., Ni, X., Han, X., & Wang, F. Y. (2019). Blockchain-enabled smart contracts: architecture, applications, and future trends. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 49(11), 2266-2277.

Wohrer, M., & Zdun, U. (2020). From Domain-Specific Language to Code: Smart Contracts and the Application of Design Patterns. IEEE Software, 37(5), 37-42.

De Sousa, V. A., Burnay, C., & Snoeck, M. (2020, June). B-MERODE: a model-driven engineering and artifact-centric approach to generate blockchain-based information systems. In International Conference on Advanced Information Systems Engineering (pp. 117-133). Springer, Cham.

Górski, T., & Bednarski, J. (2020). Applying model-driven engineering to distributed ledger deployment. IEEE Access, 8, 118245-118261.

GÓrski, T., & Bednarski, J. (2020, June). Transformation of the UML Deployment Model into a Distributed Ledger Network Configuration. In 2020 IEEE 15th International Conference of System of Systems Engineering (SoSE) (pp. 255-260). IEEE.

Lu, Q., Binh Tran, A., Weber, I., O'Connor, H., Rimba, P., Xu, X., ... & Jeffery, R. (2021). Integrated model-driven engineering of blockchain applications for business processes and asset management. Software: Practice and Experience, 51(5), 1059-1079.

Sánchez-Gómez, N., Torres-Valderrama, J., García-García, J. A., Gutiérrez, J. J., & Escalona, M. J. (2020). Model-Based Software Design and Testing in Blockchain Smart Contracts: A Systematic Literature Review. IEEE Access, 8, 164556-164569.

Hamdaqa, M., Metz, L. A. P., & Qasse, I. (2020, October). IContractML: A domain-specific language for modeling and deploying smart contracts onto multiple blockchain platforms. In Proceedings of the 12th System Analysis and Modelling Conference (pp. 34-43).

Lara, J. D., Guerra, E., & Cuadrado, J. S. (2014). When and how to use multilevel modelling. ACM Transactions on Software Engineering and Methodology (TOSEM), 24(2), 1-46.

Li, Y., Gu, P., & Zhang, C. (2014, April). Transforming UML class diagrams into HBase based on meta-model. In 2014 International Conference on Information Science, Electronics and Electrical Engineering (Vol. 2, pp. 720-724). IEEE.

Tolmach, P., Li, Y., Lin, S. W., Liu, Y., & Li, Z. (2021). A survey of smart contract formal specification and verification. ACM Computing Surveys (CSUR), 54(7), 1-38.

Syahputra, H., & Weigand, H. (2019). The development of smart contracts for heterogeneous blockchains. In Enterprise Interoperability VIII (pp. 229-238). Springer, Cham.