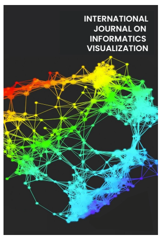
INTERNATIONAL JOURNAL
ON INFORMATICS VISUALIZATION

journal homepage : www.joiv.org/index.php/joiv



Agent-Oriented Modelling for Blockchain Application Development: Feasibility Study

Michelle Ten LiBin[a,*], Cheah WaiShiang[a], Muhammad Asyraf B Khairuddin[a], Edwin Mit[a] , Aldo Erianda[b]

[a] *Faculty of Computer Science and Information Technology, University Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia*

[b] *Department of Information Technology, Politeknik Negeri Padang, Padang, West Sumatera, Indonesia*

Corresponding author: [*]michelle10libin@gmail.com

***Abstract*— Blockchain application development has received much attention nowadays. As development is complex and challenging, a systematic approach is needed to improve the product, services, and process quality. Despite the introduction of techniques, there are still inadequate models for demonstrating the blockchain's internal architecture. Hence, there is a gap when developing the blockchain application, a gap in the modelling environment of a blockchain development application. This paper introduces a new insight into blockchain application development through Agent-Oriented Modelling (AOM). AOM is a methodology for complex socio-technical system development, and we believe that it can reduce the complexity of implementing the blockchain application. In this paper, the AOM is used to model a blockchain-based "win a fortune" system, which includes smart contract development. It showcases the feasibility of adopting AOM to model a blockchain enabling application. A usability survey among the novices has further validated the usability and benefits of AOM in the blockchain enabling application development.**

***Keywords*— Agent-oriented modelling; blockchain modelling; online games modelling.**

Manuscript received 15 Oct. 2020; revised 29 Jan. 2021; accepted 2 Feb. 2021. Date of publication 30 Sep. 2021.

I. INTRODUCTION

With the rapid growth of innovative technology, blockchain technology has received much attention nowadays. Satoshi Nakamoto found blockchain in 2008 [1]. It is a decentralized database distributed across a network of devices. It is known as a node. Blockchain provides a secure, trustworthy, and transparent environment for data transactions. Each transaction will generate a new block on the network and propagate to other nodes. The block consists of a current and previous block hash, and data are immutable once deployed to the network.

Aside from cryptocurrency, blockchain applications are growing in the most divergent fields. Regardless of the exponential surge in interest in Decentralized Applications, it currently lacks well-organized software development approaches [2]. The interest in Blockchain Technology has resulted in significant investment, generally in rapidly creating applications to be the first on the market, which has resulted in massive disasters, typically due to poor design and inadequate security practices [3]. Udokwu *et al.* [4] addressed blockchain technologies' limitations and common issues. For example, Blockchain Technology is facing complexity, usability, and architecture design issue. Hence, we have foreseen a gap in blockchain application development. Hence, the primary purpose of this paper is to introduce an alternative methodology to model blockchain applications' entire environment and show the structure of how smart contracts will be operated from an Agent-Oriented Modelling perspective.

The primary research question is how Agent-Oriented Modelling Framework can support blockchain enabling application development? The research question should answer that the agent-oriented modelling can capture blockchain characteristics and flow. Then, the sub-question is inferred based on the primary research question. What are modelling notations used to form the framework? How does the modelling notation in the framework relate to blockchain elements? What are the benefits of using the framework in the development of a blockchain enabling application? As blockchain is a complex system, Agent-Oriented Modelling (AOM) can bridge the gap. AOM is modelling that handles complex systems. With AOM, we argue that it can easily model the details of the blockchain application and the environment.

This paper presents a case study where the blockchain is used to solve a "win a fortune" game system. We used the proposed methodology, AOM, to trigger the blockchain application's functionality and potential. To evaluate the usefulness and completeness of the AOM, we experiment with a group of novice users and compare the proposed methodology with Unified Modelling Language (UML).

The remainder of the paper will be organized in the following manner: Section 1 presents the related works in engineering blockchain applications. Section 2 presents a walkthrough example of how AOM is adopted to blockchain-based lottery mobile games. Section 3 presents the analysis and findings of the adoption of AOM among novices in blockchain application development. Finally, the paper is concluded in Section 4.

A. Related Works

This section presents the review of existing researches on software development methodology for blockchain applications. First, we will categorize the existing works from the researches into few types: Software Development Life Cycle (SDLC), Agile Methodology, Agile Block Chain DApp Engineering (ABCDE), Model-Driven Methodology, Decentralized Agent-Oriented Modelling (DAOM), Action Design Research (ADR) Approach, Entity Relationship Diagram (ERD), Unified Modelling Language (UML), and Business Process Modelling Notation (BPMN).

Miraz and Ali [5] pointed out the limitations of the current methodology due to the blockchain characteristics: immutability behaviour, consensus mechanism, and gas cost during the transaction using a smart contract. Thus, Software Development Life Cycle (SDLC) models are challenging to fulfil, especially in the testing and maintenance phase. The authors summarized the difficulties of each SDLC model on blockchain applications. The approaches (Waterfall Model, V-shape Model, Iteration and Incremental Model, Spiral Model, Big Band Model, and Agile Model) that the authors mention require the user's involvement to give feedback in the testing phase is impractical. Thus, testing the blockchain requires a high volume of transactions for reaching a consensus for verification and validation works. Besides, maintenance for future improvement is impossible on blockchain applications due to its immutability behaviour. Big Bang Model is the most unsuitable model for blockchain, leading to risks and bugs without proper planning.

Fridgen et al. [6] addressed that blockchain requires a technology-driven and systematic approach to design and develop potential blockchain use cases. The researchers apply the action design research (ADR) approach and situational method engineering (SME) to develop blockchain use cases. ADR consists of four steps: "problem formulation", "building, intervention, and evaluation", "reflection and learning", and "formalization of learning".

According to [7], there is still a lack of modelling notation for blockchain technology. Therefore, the main objective of the research is to raise attention to blockchain modelling notation. From the review, the authors adopt Entity Relationship Diagram (ERD), Unified Modelling Language (UML), and Business Process Model Notation (BPMN) to develop Blockchain-oriented Software (BOS).

Researchers proposed adopting the Agile approach, a flexible, iterative, and adaptive methodology that involves gathering requirements, development, testing, and gathering feedback from users and enhancement. Agile methodology allows us to ensure software quality and usability. This methodology has been applied in various projects such as the Crypto-Trading project [8], smart cities project [9], Initial Coin Offering [10], and Supply Chain Operations [11]. Researchers proposed an Agile approach with a flexible, iterative, and adaptive methodology that involves gathering requirements, developing, testing, and gathering feedback from users and enhancement.

Since the year 2018, a team of researchers has made a great effort to propose an extension methodology from the agile approach, known as Agile Block Chain Development Engineering (ABCDE) [3], [12]. The idea behind using ABCDE is because the smart contract is complicated. Unlike traditional programming, smart contracts introduced new concepts such as the contract's address to perform a transaction; gas requires executing the contract and digital money transfer between two parties [12]. ABCDE follows Agile approach practices such as user stories and iterative development processes. However, ABCDE will split the development process into two: front-end development and smart contract development. The contribution of the ABCDE methodology triggers the developers to implement on the DApp project such as Food Supply Chain [13], worker contracts management [2], and Food Distribution Chain [14]. The advantages of this methodology help to prescribe the actors and user stories from the beginning, which helps simplify the system and smart contract configurations. Besides, this action will help lower the chances of failure since the modelling notations highlighted all the important actors, requirements, and connections.

On the other hand, Model-driven engineering (MDE) is widely used in blockchain technology [15], [16], [17], and [18]. Model-Driven is convenient to design models and convert them to code by using tools. [17] pointed out that the code generation tools allowed us to apply best practices and well-tested building blocks.

The study in [19] introduces decentralized agent-oriented modelling (DAOM) based on the model-driven framework. These models accommodate both technical and non-technical clients' communications. It is a three-step procedure for determining requirements and describing the architecture of a DApp. First, the AOM goal model is used in the initial stage to define the preliminary needs for dealing with the usability and complexity issues. The second stage involves connecting the UML component diagram to the project architecture. Finally, a UML sequence diagram is used to support the protocols and blockchain operations.

From the review, modelling is needed to model the complexity of blockchain enabling application development and to prevent bugs and errors. Furthermore, the smart contract will be implemented on various applications that handle sensitive data and money, so it is essential to have a formalized model for a systematic way to plan, implement, and design the blockchain application.

In order to fill this gap, Agent-Oriented Modelling (AOM) is proposed and used in our research. The benefits of using AOM are to document how the agent achieves the goal and how the agent behaves in a different situation, as elaborated in the following section. To date, AOM has been adopted in 3D simulation [19], video surveillance [20], emotion application [21], games [22].

II. Material and Method

A walkthrough case study is demonstrated to showcase the feasibility of AOM in the blockchain enabling application development. The case study is known as win a fortune games. To win a fortune is a type of game that provides players with the opportunity to win big prizes. Some people's odds to play a "win a fortune" game to win an enormous fortune, some are for fun. However, there is a trust issue between the win a fortune industry and the traditional win a fortune game players. In the game, the industry would question the deal and ticket from the winner real? Is the winner real? The most concern question by the players is the prized paid on time? Therefore, blockchain technology is able to solve the trust issue. By adopting blockchain technology to the "win a fortune" industry, players will buy with confidence since most smart contracts are public. Players can check all the rules and agreements before the bet.

The "win a fortune" smart contract will be implemented using the Ethereum platform. There are two main parties to operate the lottery smart contract: the manager and the players. The manager is responsible for deploying the smart contract, announcing the game rules, checking the amount of the contract Ethers, and selecting one winner from the players. Win a fortune game start when a player begins the bet by sending their Ethers through their wallet (MetaMask, Mist, and MyEther) to the contract. Players can decide on their luck by pay with a minimum of 0.01 Ethers but without the maximum limit to gameplay games. Each player who enters the game will be added to an array list, and the contract will hold the amount of the Ethers sent by the players. The manager is allowed to check the total amount of the Ethers from all of the players. Once the times up, the "win a fortune" game manager will execute the contract to select a winner. In the contract, will first generate a random number from the array list and pick a winner. Lastly, the total amount of Ethers in the contract will automatically transfer to the winner's wallet. Therefore, the prizes won by the winner are assured. Then the array list will be reset to prepare for the next game.

As we were known, AOM has three phases that cover the conceptualization layer, the specification layer, and the realization layer. The first layer will help to identify the problem and motivation from the human perspective. The models in the conceptualization layer are the goal model, role model, organization model, and domain model. The specification layer will then include the system's design using the knowledge, interaction, scenario, and behaviour models. Then lastly, the realization layer will lead to decide the platform adoption.

AOM is started with goal modelling as Figure 1 illustrates the goal of operating a lottery company consisting of reliability, integrity, ethics, and scalability. The main person in charge of the lottery company is the manager. To manage a lottery company, the manager must set and obey the rules and regulations that need transparency. The company should locate a sales target and achieve it by selling the ticket to the players. Lastly, the manager should pick a winner from the player in a fair mode. The winner will show the lottery ticket to the manager and perform verification and validation. Once the validation passed, the prize is given to the winner.

Figure 2 shows the goal of a player to win a fortune. To win a big prize, the player must have faith and integrity to join a bet. The player must buy a ticket and obey the rules and regulations stated by the lottery company. If the player is selected as the winner, the winner must show his/her lottery ticket as proof to redeem the prizes.

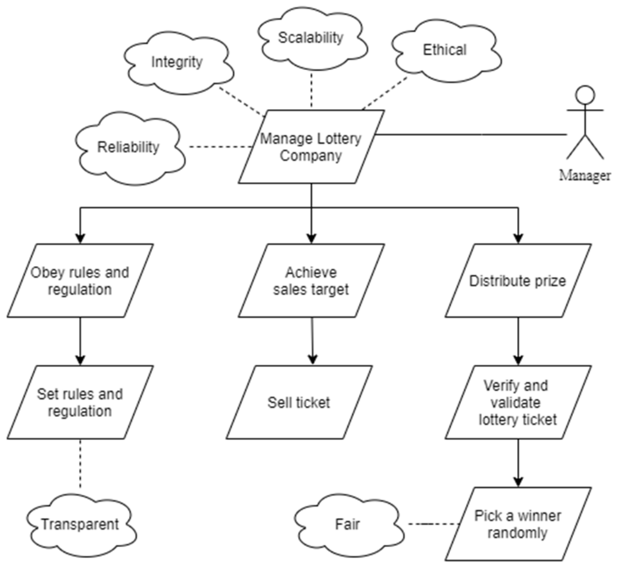


Fig. 1 Goal Model for Manage Lottery Company

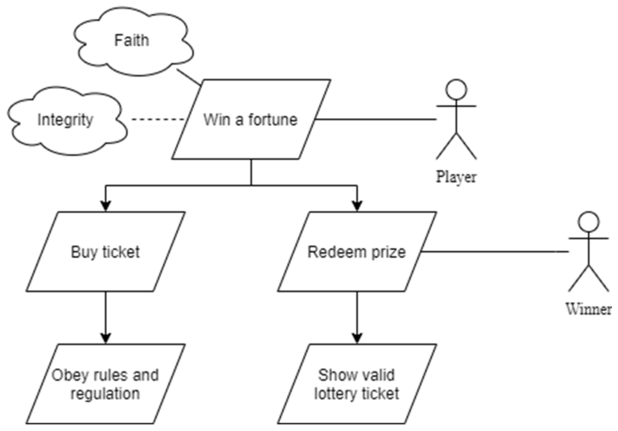


Fig. 2 Goal Model for Win a Fortune

The roles are included, manager and player. The task description of each role is shown in Table 1.

TABLE I
A ROLE MODEL FOR MANAGER AND PLAYER

Name	Manager
Description	• A person that manages the operation of the company.
Responsibility	• Set, announce and obey rules and regulations. • Achieve sales by selling the ticket. • To a select winner. • Verify and validate the winning lottery ticket. • Distribute the prize to the winner.
Constraint	• Select winner based on the rules and regulations stated.

Name	Player
Description	• A person that wishes to win a fortune by buying a lottery.
Responsibility	• Buy a lottery ticket at the official lottery counter. • Wait for the lottery announcement for the winner.
Constraint	• Must purchase a ticket from the official counter.

The organizational model in Figure 3 shows the relationship between the manager, player, and winner. The organizational model shows that the manager knows the player. Therefore, he/she will select one winner from the players.

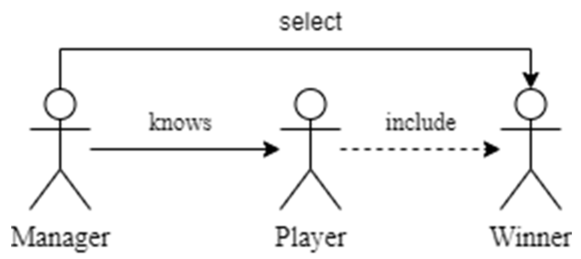


Fig. 3 Organizational Model for Lottery Smart Contract

Figure 4 shows the domain model for the Lottery case study. The manager is responsible for setting the rules and regulations of the company, which are required every player to comply. Then the player may make payment and buy a ticket from the manager. When the game end, the manager will select one winner from the player. The winner needs to show the ticket to the manager. Once the manager verifies the ticket, the prize will be handed to the winner.

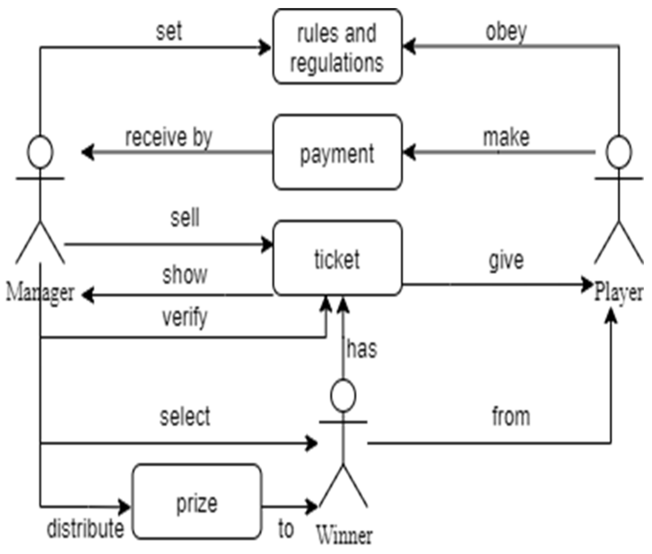


Fig. 4 Domain Model for Lottery Company

Figure 5 illustrates the knowledge models that represent the private and shared knowledge among the agents. The human agents (manager, player, and winner) will have their wallet details such as public key, private key, and account balance, which should be kept private. Human agents participating in this lottery web application should know the programmatically coded rules in the smart contract. The rules included the game state, checking the manager address that returns in Boolean, the minimum bet, the prize distribution, and the random number algorithm. Once the players join the game, the player address is added to the ticket array, visible among the players and the manager. The report in the smart contract is kept private, which contains the contract balance from the player. Lastly, the result will only be revealed to the winner.

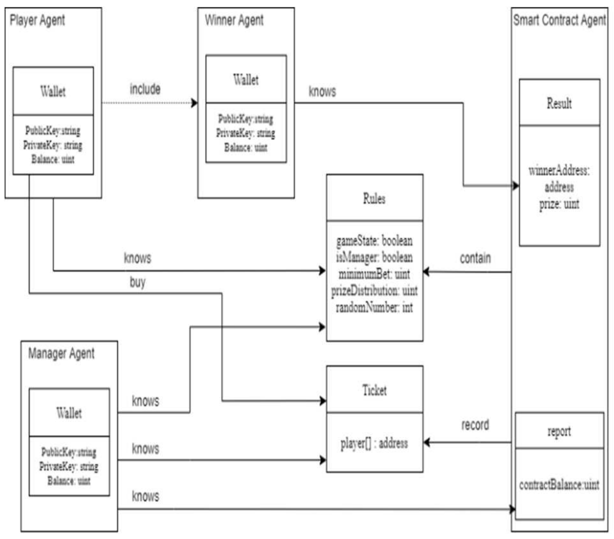


Fig. 5 Knowledge Model for Lottery Smart Contract

The scenario model will explain the sequence steps to achieve the goals of the lottery smart contract. Table 2 shows the scenario model for prize distribution to the player.

Figure 6 shows the interaction when the manager selects the winner; Figure 7 shows the behaviour model when the player plays the game, and Figure 8 shows the behaviour model when the manager is ready to select a winner. In Figure 8, R1 is trigger when the game state is true. When the times up, the manager could select a winner, which will help to trigger the wallet. This action will invoke the smart contract. R2 will be performed when the game state is false and check whether the manager triggers the smart contract. If yes, the smart contract will loop through each player's address and get a random number indicating the array's index. Next, the smart contract will get the winner's address and the contract balance to transfer the prizes. Lastly, the smart contract will clear the player list and set the game state to true for the next game.

TABLE II
A SCENARIO MODEL

Scenario 1				
Quality Goal	Integrity			
Initiator	Manager Agent			
Trigger	Distribute prize			
Description				
Condition	**Step**	**Activity**	**Agent types/role**	**Resources**
	1	Visit manager portal	Manager agent	-
	2	Select winner	Manager agent	Web
	3	Connect wallet	Manager agent	-
	4	Request signature	Manager agent	Wallet
	5	Provide signature	Manager agent	Manager
	6	Trigger smart contract	Web 3 agent	-
	7	Check address (is the manager)	Smart contract agent	Blockchain
The smart contract is trigger by the manager	8	Loop player address	Smart contract agent	Blockchain
	9	Random select one winner	Smart contract agent	Smart contract
	10	Get winner address		Blockchain
	11	Get contract balance	Smart contract agent	Blockchain
	12	Distribute prize to the winner	Smart contract agent	Smart contract
	13	Reset player list		Smart contract
Game state = true	14	Prepare for the next game		Smart contract

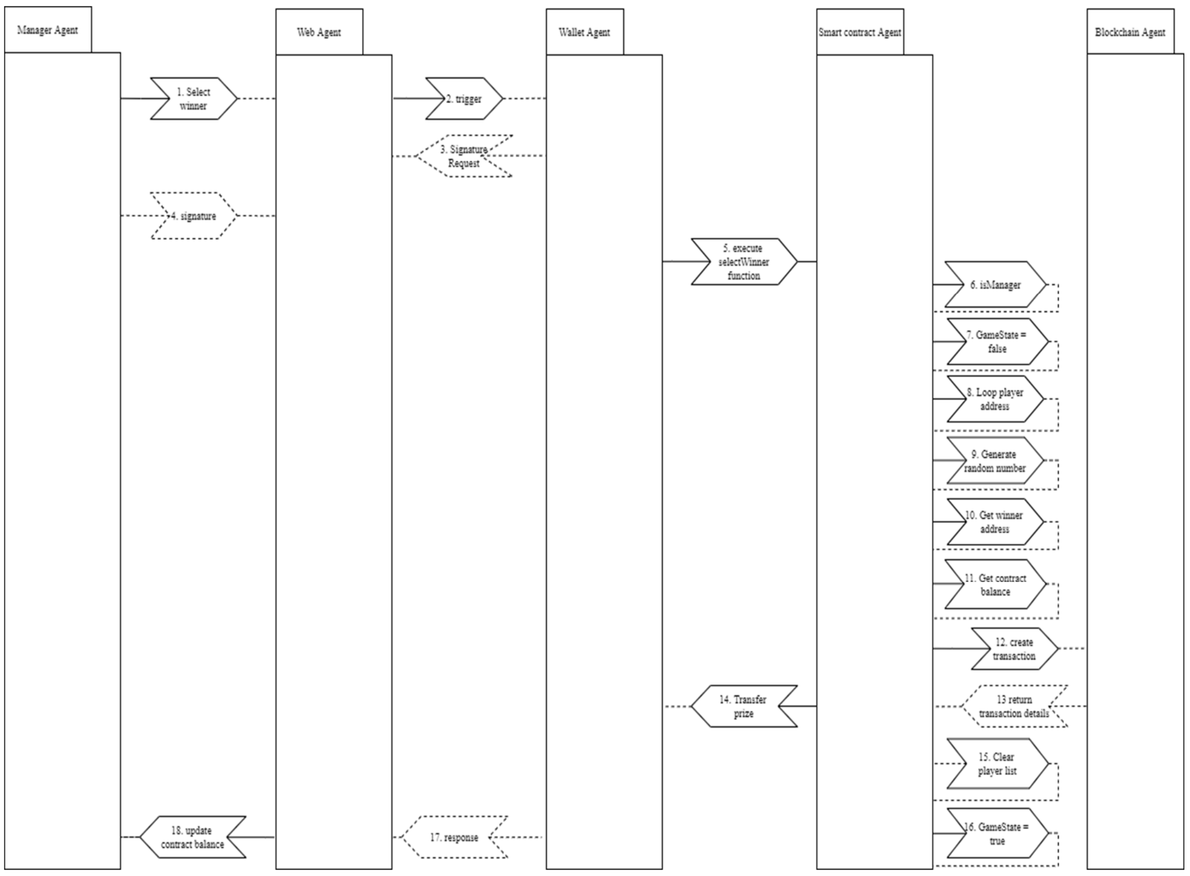


Fig. 6 Interaction Model for Lottery Smart Contract Deployment

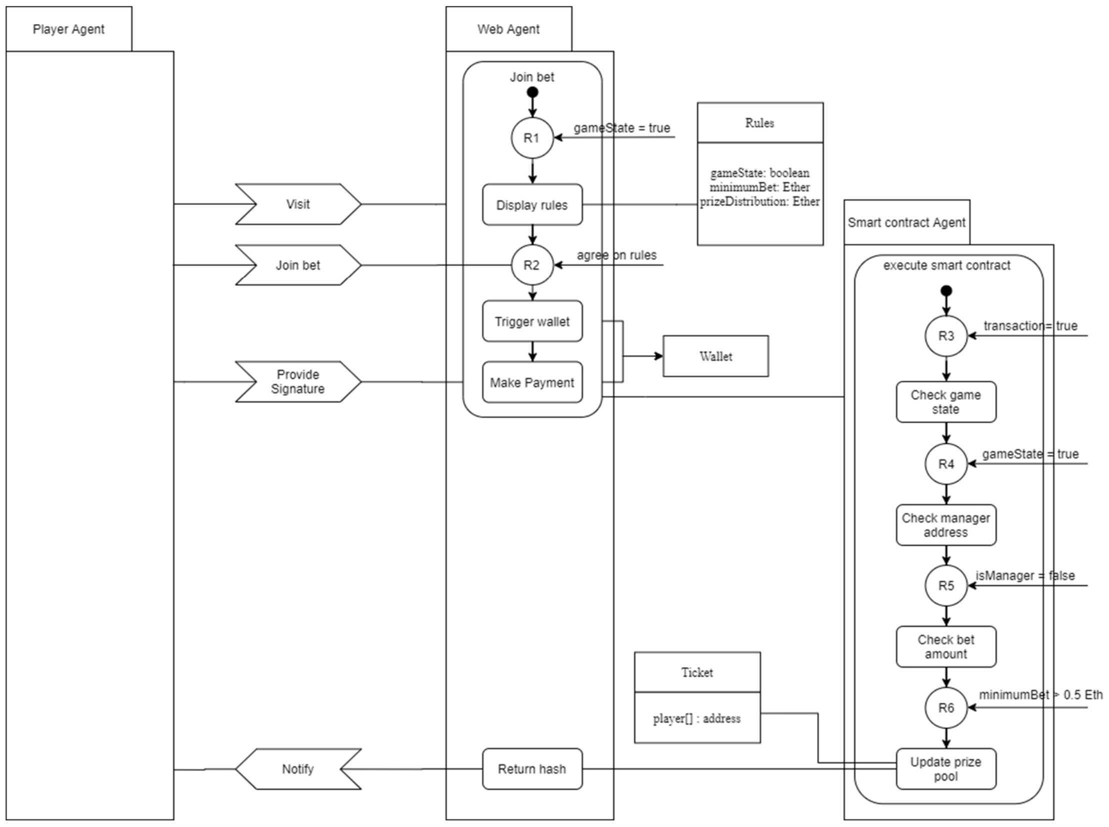


Fig. 7 Behaviour Model for Lottery Smart Contract

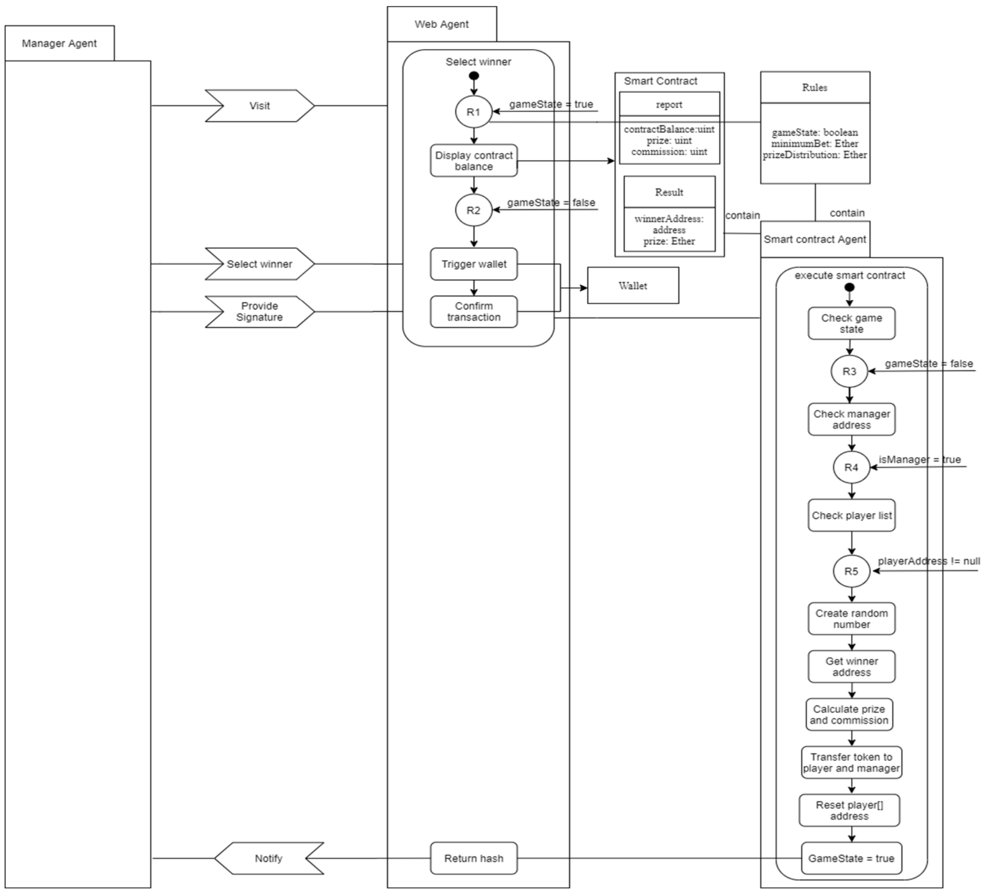


Fig. 8 Behaviour Model for Lottery Smart Contract

III. Results and Discussion

To evaluate the Agent-Oriented Modelling for Blockchain development, we conduct an experiment among novices without any blockchain knowledge or background. The novices are selected from bachelor students in software engineering with a firm knowledge of UML and AOM. In addition, the students have taken a course, Advance Software Engineering, for a semester.

Throughout this course, a fundamental of blockchain knowledge is delivered to the students through lectures. To validate the outcome, the students must work on an assignment to present their blockchain knowledge and the existing methodologies through the recently published journal.

For the first stage of the experiment, the students are given a task to model the blockchain application using UML based on the case study. In the second stage, the students are required to model the same case study by using AOM. Then, the models are evaluated based on the requirements as shown in Table 3, which are extracted from the case study.

There are 50 students' models reviewed and evaluated based on the requirements, and each requirement is given one point. The students that attended the lecture shared a similar grasp of the Blockchain concept. They used the same understanding of blockchain technology to apply to the UML and AOM. However, the ability of each student to model the case study is different, which produce the result in Table 4. The result for AOM is higher than UML. In other words, AOM has higher possibilities and potential to models the requirements of blockchain technology.

TABLE III
REQUIREMENTS BASED ON THE CASE STUDY

No	As a …	Will...	Leads to…
1	Player	able to check the rules and regulations before bet	Transparent
2	Player	sends ether through wallet	Cryptocurrency
3	Smart contract	checks the minimum ether	Autonomous
4	Smart contract	handle a list of player address	Immutable
5	Smart contract	handle ether sent by the player	Immutable
6	Manager	can check the amount of the contract	Authentication
7	Manager	execute smart contract to select a winner	Autonomous
8	Smart contract	generates a random number from the player list	Fairness, autonomous, and trustworthy
9	Smart contract	Transfer prize to winner automatically	Autonomous and trustworthy
10	Smart contract	clear array list and reset the game	Autonomous

From the UML model, the requirements did successfully captured on the use case model. The class diagram manages to handle the data required of the actors, and the sequence diagram shows the interaction on every event. However, the activity diagram is not detailed enough to capture the flow of the actors.

TABLE IV
STUDENTS MODELS RESULT BASED ON THE REQUIREMENTS

No	Requirement	UML	AOM
1	Players able to check the rules and regulations before bet	38.0 %	28.0 %
2	Player sends ether through wallet	76.0 %	92.0 %
3	Smart contract checks the minimum ether	24.0 %	36.0 %
4	Smart contract handle list of player address	48.0 %	64.0 %
5	Smart contract handle ether sent by the player	78.0 %	70.0 %
6	Only the manager can check the amount of the contract	48.0 %	74.0 %
7	Manager executes smart contract to select a winner	68.0 %	68.0 %
8	Smart contract generates a random number from the player list	58.0 %	78.0 %
9	The prize is automatically sent to the winner	68.0 %	74.0 %
10	Smart contract clear array list and reset the game	46.0 %	32.0 %
	Total	53.2%	62.6%

Based on their AOM model, it is interesting to report that the goal model did reflect the blockchain application development the most because it stated the main and sub-goal, and the characteristics of blockchain are stated using the functional goal. The role model will then explain the agent's role to help programmers and stakeholders understand industry-specific knowledge for each role. Nevertheless, most students focus on the human agents without the smart contract agent because they treat the smart contract as a class instead of an agent. The interaction models from the students cannot show how a smart contract is triggered through a transaction.

Lastly, a survey is conducted to understand the student opinion regarding using AOM in modelling a blockchain enabling application like "win a fortune" games. 68 students had responded to the survey, and the analysis result is as shown in Figure 8. There are 8 of the students who strongly agree that AOM can reflect on blockchain application development; 32 of the students agree due to the completeness of the model and ease of understanding by the stakeholders and programmer. On the other hand, 19 students rated neutral as most of them lack blockchain technology knowledge; 9 students disagreed with AOM able to reflect blockchain application development and feel UML is more suitable to model smart contracts as it acts as a class.

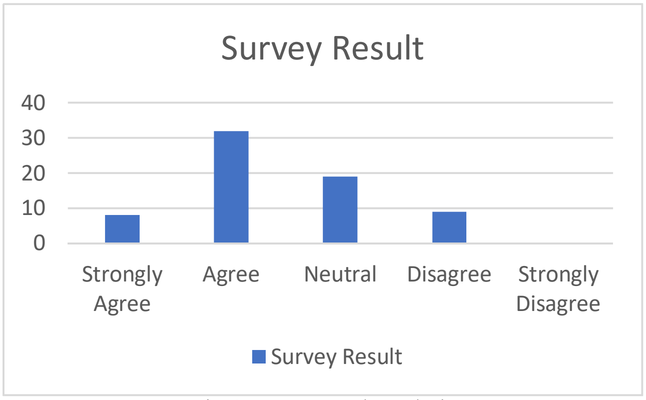


Fig. 8 Survey Result Analysis

It is still difficult to differentiate to what extent we need to develop or introduce blockchain elements in the entire system from evaluating students' works. Do we need to develop it immediately given the problem or system? When do we need to deploy blockchain, and what are the benefits of it? Although the AOM can model the concept of blockchain, the participants feel it is challenging to model it due to lacking deep knowledge of blockchain technology.

From the survey, it is interested in reporting that blockchain development is a complex process. Without in-depth knowledge of blockchain, novices seem no clue where to start on it. For this reason, we shall need a methodology to lead the novices to produce a potential blockchain application by using AOM, as it seems that AOM is aided to extract more blockchain requirements while analyzing and understanding the use case problem.

IV. Conclusion

The main goal of this paper is to introduce Agent-Oriented Modelling to solve the complexity and usability issues of blockchain enabling applications. To achieve this, we first apply the Agent-Oriented Modelling to a "win a fortune" case study. Then, we evaluate the proposed methodology with a group of Software Engineering students.

The first step of the methodology introduces planning and discussion on the requirements' details between stakeholders and programmers until smart contracts are deployed through agent models in this paper. In the goal, the model illustrates the motivation of the system from a human perspective. Role models describe the responsibilities and task description involved in the system, and the Organizational model shows the relationship between the roles. The domain model shows how the roles will handle the information of the system in any environment. From the scenario model, each task is explained in a sequence order to achieve the goal. We can view the interaction and the information exchange between the agents from the interaction model in more detail. The knowledge model shows the shared and private knowledge of the agents in the system. Lastly, the overview of the entire development is shown in the behaviour model. By adopting AOM to represent the blockchain application, we believe that the blockchain developer can get familiar with the lifecycle and the requirements of blockchain technology. More works are needed further to verify the correctness and usability of the AOM, especially how AOM can rapidly prototype the blockchain enabling application.

The future work for this study is to extend the Agent-Oriented Modelling by adding modelling notation to explore further the potential and adaptability of the blockchain on an application. The current methodology mainly focuses on the analysis and design of the application. Therefore, the future work proposes extending the current methodology to analyze and understand the use case problem and highlight the needs of blockchain technology to convince the stakeholder of the project to adopt blockchain.

Acknowledgement

We would like to thank UNIMAS to support this research.

References

[1] S. Nakamoto, "Bitcoin : A Peer-to-Peer Electronic Cash System," pp. 1–9, 2008.

[2] G. Lallai, A. Pinna, M. Marchesi, and R. Tonelli, "Software Engineering for DApp Smart Contracts managing workers Contracts Software Engineering for DApp Smart Contracts managing workers Contracts," no. April, 2020.

[3] L. Marchesi, M. Marchesi, and R. Tonelli, "An Agile Software Engineering Method to Design Blockchain Applications," no. Secr, 2018.

[4] C. Udokwu, A. Kormiltsyn, K. Thangalimodzi, and A. Norta, "The State of the Art for Blockchain-Enabled Smart-Contract Applications in the Organization," *Proc. - 2018 Ivannikov Isp Ras Open Conf. ISPRAS 2018*, no. November, pp. 137–144, 2019.

[5] Miraz, M. H., & Ali, M. (2020). Blockchain Enabled Smart Contract Based Applications: Deficiencies with the Software Development Life Cycle Models. 33(1), 101–116. Retrieved from http://arxiv.org/abs/2001.10589.

[6] G. Fridgen, J. Lockl, S. Radszuwill, A. Rieger, A. Schweizer, and N. Urbach, "A solution in search of a problem: A method for the development of blockchain use cases," *Am. Conf. Inf. Syst. 2018 Digit. Disruption, AMCIS 2018*, vol. 4801, 2018.

[7] H. Rocha and S. Ducasse, "Preliminary Steps Towards Modeling Blockchain Oriented Software," *2018 IEEE/ACM 1st Int. Work. Emerg. Trends Softw. Eng. Blockchain*, pp. 52–57, 2018.

[8] K. Mannaro, A. Pinna, and M. Marchesi, "Crypto-Trading: Blockchain-oriented energy market," *2017 AEIT Int. Annu. Conf. Infrastructures Energy ICT Oppor. Foster. Innov. AEIT 2017*, vol. 2017-Janua, pp. 1–5, 2017.

[9] S. Ibba, A. Pinna, and F. E. Pani, "CitySense : blockchain-oriented Smart Cities," 2017.

[10] S. Ibba, A. Pinna, M. I. Lunesu, M. Marchesi, and R. Tonelli, "Initial coin offerings and agile practices," *Futur. Internet*, vol. 10, no. 11, pp. 1–21, 2018.

[11] J. M. Montes, V. M. Larios, M. Avalos, and C. E. Ramírez, "Applying Blockchain to Supply Chain Operations at IBM Implementing Agile Practices in a Smart City Environment," *Res. Comput. Sci.*, vol. 147, no. 2, pp. 65–75, 2018.

[12] L. Marchesi, M. Marchesi, and R. Tonelli, "ABCDE – Agile Block Chain Dapp Engineering," no. December, 2019.

[13] G. Baralla, A. Pinna, and G. Corrias, "Ensure Traceability in European Food Supply Chain by using a blockchain Ensure Traceability in European Food Supply Chain by using a blockchain System," no. March, 2019.

[14] A. Pinna, M. Marchesi, and S. Ibba, "Ensuring transparency and traceability of food local products : A blockchain application to a Smart T ourism Region," no. April, pp. 1–18, 2020.

[15] C. K. Frantz and M. Nowostawski, "From Institutions to Code : Towards Automated Generation of Smart Contracts From Institutions to Code : Towards Automated Generation of Smart Contracts," no. November 2017, 2016.

[16] S. Seebacher and M. Maleshkova, "A Model-driven Approach for the Description of Blockchain Business Networks A Model-driven Approach for the Description of Blockchain Business Networks," no. January, 2018.

[17] C. Di Ciccio *et al.*, "Blockchain Support for Collaborative Business Processes," *Informatik-Spektrum*, vol. 42, no. 3, pp. 182–190, 2019.

[18] M. Jurgelaitis, R. Butkienė, E. Vaičiukynas, V. Drungilas, and L. Čeponienė, "Modelling principles for blockchain-based implementation of business or scientific processes," *CEUR Workshop Proc.*, vol. 2470, pp. 43–47, 2019.

[19] C. Udokwu, H. Anyanka, and A. Norta, "Evaluation of Approaches for Designing and Developing Decentralized Applications on Blockchain," pp. 55–62, 2020.

[20] C. W. Shiang, B. T. Onn, F. S. Tee, M. A. Bin Khairuddin, and M. Mahunnah, "Developing agent-oriented video surveillance system through agent-oriented methodology (AOM)," *J. Comput. Inf. Technol.*, vol. 24, no. 4, pp. 349–368, 2016.

[21] S. F. Zulkifli, C. W. Shiang, M. A. bin Khairuddin, and N. bt Jali, "Modeling emotion oriented approach through agent-oriented approach," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 10, no. 2, pp. 647–653, 2020.

[22] M. Baldoni, C. Baroglio, D. Calvanese, and R. Micalizio, "Data and Norm-aware Multiagent Systems for Software Modularization (Position Paper)," *Eng. Multi-Agent,* 2016.