



Faculty of Computer Science and Information Technology

**An Extended Methodology Based on Agent-Oriented Modelling
Approach for Blockchain Application Development**

Michelle Ten LiBin

**Master of Science
2023**

An Extended Methodology Based on Agent-Oriented Modelling Approach for Blockchain Application Development

Michelle Ten LiBin

A thesis submitted

In fulfillment of the requirements for the degree of Master of Science

(Software Engineering)

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2023

DECLARATION

I declare that the work in this thesis was carried out in accordance with the regulations of Universiti Malaysia Sarawak. Except where due acknowledgements have been made, the work is that of the author alone. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.



.....

Signature

Name: Michelle Ten LiBin

Matric No.: 19020109

Faculty of Computer Science and Information Technology

Universiti Malaysia Sarawak

Date:1/5/2023

ACKNOWLEDGEMENT

Undertaking this master's research has been a truly life-changing experience for me, and it would not have been possible without the support and guidance that I received from many people.

First and foremost, I would like to express my special thanks to my supervisor Dr Cheah Wai Shiang.

I gratefully acknowledge the funding from the Zamalah Siswazah UNIMAS Scholarship to undertake my Master. Thanks to Associate Professor Dr Chiew Kang Leng to give me this opportunity to receive the scholarship.

I want to express my gratitude to the Centre for Graduate Studies and the administration of Universiti Malaysia Sarawak for guidance and support.

Lastly, my family deserves endless gratitude: my father, Ten Eng Le; my mother Cecilia Lee Phaik See; and my siblings, Melanie Ten Li Yen and Melisa Ten Li Lian, who have been supporting me along my master's journey.

ABSTRACT

Blockchain application development has received much attention nowadays. Blockchain handles sensitive data, which requires a layer of complexity such as cryptography, consensus mechanism, transaction immutability, and peer-to-peer network to manage decentralised data features, making blockchain development complex and challenging. Smart contracts are self-executing digital contracts that are stored on a blockchain. They enable parties to transact with each other without the need for intermediaries, making transactions faster, cheaper, and more secure. Through the synergy of blockchain and smart contracts, it is possible to create decentralised applications that can be used in a wide range of industries, bringing increased transparency, security, and efficiency to traditional business models. Various software methodologies have been introduced to support the development of blockchain applications systematically as well as reduce the development complexity. Despite the introduction of methodologies, further refinements are required to enhance their effectiveness for modelling. Moreover, all models are generic and do not consider the blockchain concept as first-class entities. Hence, a gap exists in extending and transforming the current blockchain modelling and practices when developing a blockchain application. This research introduces a new insight into blockchain-based application development through extended Agent-Oriented Modelling (AOM) to discover blockchain opportunities and commit to the growth of the technology. AOM is a methodology for complex socio-technical system development and can become an alternative methodology for blockchain application development. A walkthrough example of how extended AOM is used to model a blockchain-based lottery application is presented in this thesis. It reveals the issues and limitations of AOM, an extension of AOM, and the potential of extended AOM and turns it into a software methodology for blockchain-based application development. Furthermore,

the extension of AOM is discussed to deal with the software challenges the novices faced. Three experiments were conducted with the novices. The novices (e.g. students) were required to construct the case study using Unified Modelling Language (UML), AOM, and extended AOM. The first experiment aimed to compare UML and AOM, which resulted in a mean mark of 15.636 for UML and 17.000 for AOM. The result revealed the potential usage of AOM to model blockchain applications apart from UML. The second experiment was targeted to specify the limitation of AOM in blockchain application development. From the findings, 78% of the students failed to model blockchain requirements through AOM because it needs to be more specific to identify blockchain needs. Hence, an extended AOM is needed. The third experiment intended to test the usability of the extended AOM and AOM. 44% of the students managed to capture more than 80% of blockchain requirements through extended AOM. On the other hand, 51% of the students scored between 20% to 60% on the blockchain requirement, and only 5% failed to model blockchain requirements through extended AOM. The results reveal the usage of the extended AOM to model blockchain-enabled applications comprehensively.

Keywords: Blockchain, smart contract, methodology, agent-oriented modelling, Ethereum

Metodologi Diperluaskan Berdasarkan Permodelan Berorientasikan Ejen untuk Pembangunan Aplikasi Blockchain

ABSTRAK

Pembangunan aplikasi blockchain (blok rantai) makin jadi perhatian pada masa kini. Blockchain menyimpan data sensitif yang dilakukan dengan mata wang kripto, algoritma konsensus, transaksi, peer to peer dan terdesentralisasi. Kontrak pintar adalah kontrak digital yang dapat melaksanakan diri sendiri dan disimpan pada blockchain. Mereka membolehkan pihak-pihak bertransaksi dengan satu sama lain tanpa memerlukan pihak tengah, menjadikan transaksi lebih cepat, lebih murah, dan lebih selamat. Dengan gabungan penggunaan teknologi blockchain dan kontrak pintar, ia dapat mencipta aplikasi terdesentralisasi yang boleh digunakan dalam pelbagai industri, untuk menaikkan taraf keselamatan, dan kecekapan kepada model perniagaan tradisional. Pelbagai metodologi perisian telah diperkenalkan untuk menyokong pembangunan aplikasi blockchain secara sistematik dan mengurangkan kerumitan pembangunan. Walaupun metodologi telah diperkenalkan, taraf kesan modelling masih perlu dipertingkatkan. Selain itu, semua model adalah generik dan tidak menganggap konsep blockchain sebagai entiti kelas pertama. Oleh itu, jurang wujud dalam memanjangkan dan mengubah permodelan dan amalan blockchain semasa semasa membangunkan aplikasi blockchain. Penyelidikan ini memperkenalkan pandangan baharu ke dalam pembangunan aplikasi berasaskan rangkaian blok melalui Permodelan Berorientasikan Ejen (AOM) lanjutan untuk menemui peluang rangkaian blok dan komited terhadap pertumbuhan teknologi. AOM ialah metodologi untuk pembangunan sistem sosial-teknikal yang kompleks dan boleh menjadi metodologi alternatif untuk pembangunan aplikasi blockchain. Contoh panduan tentang cara lanjutan AOM digunakan untuk memodelkan aplikasi loteri berasaskan blockchain dibentangkan dalam tesis ini. Ia

mendedahkan isu dan had AOM, lanjutan AOM, dan potensi AOM lanjutan dan mengubahnya menjadi metodologi perisian untuk pembangunan aplikasi berasaskan blockchain. Tambahan pula, lanjutan AOM dibincangkan untuk menangani cabaran perisian yang dihadapi oleh pemula. Tiga eksperimen telah dijalankan dengan orang baru. Orang baru (contohnya pelajar) dikehendaki membina kajian kes menggunakan Bahasa Pemodelan Bersepadu (UML), AOM dan AOM lanjutan. Eksperimen pertama bertujuan untuk membandingkan UML dan AOM, yang menghasilkan markah min 15.636 untuk UML dan 17.000 untuk AOM. Hasil eksperimen mendedahkan potensi penggunaan AOM untuk memodelkan aplikasi blockchain selain daripada UML. Eksperimen kedua disasarkan untuk menentukan had AOM dalam pembangunan aplikasi blockchain. Daripada penemuan, 78% daripada jumlah pelajar gagal memodelkan keperluan blockchain melalui AOM kerana ia perlu lebih khusus untuk mengenal pasti keperluan blockchain. Oleh itu, AOM Lanjutan diperlukan. Eksperimen ketiga bertujuan untuk menguji kebolehgunaan AOM dan AOM Lanjutan. 44% daripada jumlah pelajar berjaya menangkap lebih daripada 80% keperluan blockchain melalui AOM Lanjutan. Sebaliknya, 51% daripada jumlah pelajar mendapat markah antara 20% hingga 60% pada keperluan blockchain, dan hanya 5% gagal memodelkan keperluan blockchain melalui AOM Lanjutan. Hasilnya mendedahkan penggunaan AOM Lanjutan untuk memodelkan aplikasi yang didayakan blockchain secara menyeluruh.

Kata kunci: Blok Rantai, kontrak pintar, metodologi, permodelan berorientasikan ejen, Ethereum

TABLE OF CONTENTS

	Page
DECLARATION	i
ACKNOWLEDGEMENT	ii
ABSTRACT	iii
<i>ABSTRAK</i>	v
TABLE OF CONTENTS	vii
LIST OF TABLES	xi
LIST OF FIGURES	xiii
LIST OF ABBREVIATIONS	xv
CHAPTER 1: INTRODUCTION	1
1.1 Introduction	1
1.2 Problem Statement	6
1.3 Research Objectives	7
1.4 Research Questions	8
1.5 Scope of the Research	8
1.6 Expected Outcome	9
1.7 Thesis Structure	9
CHAPTER 2: LITERATURE REVIEW	11
2.1 Overview	11

2.2	Software Methodologies for Blockchain Application Development	11
2.2.1	Software Development Life Cycle	17
2.2.2	Action Design Research and Situational Method Engineering	17
2.2.3	General Agile Approach	18
2.2.4	Agile Block Chain Development Engineering (ABCDE)	20
2.2.5	Model-Driven Engineering (MDE)	24
2.2.6	Agent Oriented Modelling	27
2.2.7	<i>e3</i> Value Modelling	29
2.3	Discussion	30
2.4	Introduction of Agent Oriented Modelling and <i>e3</i> value model	36
2.4.1	Elicit Requirements	36
2.4.2	Modelling	38
2.5	Chapter Summary	43
	CHAPTER 3: PROPOSED EXTENDED METHODOLOGY	44
3.1	Overview	44
3.2	Research Methodology	44
3.3	Rationale in Extending the AOM	45
3.4	Extended Agent-Oriented Methodology	45
3.4.1	Phase 1: Requirement Elicitation Layer	48
3.4.2	Phase 2: Computation Independent Modelling	51

3.4.3	Phase 3: Early Identification of Blockchain Use Case	59
3.4.4	Phase 4: Platform Independent Modelling (PIM)	63
3.4.5	Phase 5: Platform Specific Model	69
3.5	Chapter Summary	71
CHAPTER 4: PROTOTYPING AND PILOT TESTING OF THE EXTENDED AOM		72
4.1	Overview	72
4.2	Elaborate Case Study	72
4.3	Modelling of Blockchain-based Lottery Application through Extended AOM	73
4.3.1	Phase 1: Requirement Elicitation of Blockchain-based Lottery Application	73
4.3.2	Phase 2: Computation Independent Modelling	77
4.3.3	Phase 3: Early Identification of Blockchain Use Case	84
4.3.4	Phase 4: Platform Independent Design and Modelling	89
4.3.5	Phase 5: Platform Specific Model	97
4.4	Lesson Learnt	102
4.5	Conclusion	104
CHAPTER 5: EVALUATION OF THE EXTENDED AOM		105
5.1	Overview	105
5.2	Experiment Setting	105
5.2.1	Students' Background	108

5.3	Experiment 1 – The Usage of UML and AOM for Blockchain Application Development	109
5.3.1	The Walkthrough of the Lottery Case Study for Experiment 1	109
5.3.2	Data Analysis for Experiment 1	111
5.4	Experiment 2 – The Usage of AOM for Blockchain Application Development	114
5.4.1	The Walkthrough of the Lottery Case Study for Experiment 2 and Experiment 3	115
5.4.2	Scoring System	118
5.4.3	Result Analysis on Experiment 2	119
5.5	Experiment 3 – The Usage of Extended AOM for Blockchain Application Development	123
5.6	Discussion	132
5.7	Threats to Validity	133
5.8	Conclusion	133
	CHAPTER 6: CONCLUSION	135
6.1	Overview	135
6.2	Contributions	135
6.3	Limitations and Future Work	136
6.4	Conclusion	137
	REFERENCES	138
	APPENDICES	147

LIST OF TABLES

	Page
Table 2.1 Software Methodologies for Blockchain Application Development	13
Table 2.2 Strength and Weakness of Existing Methodology	32
Table 2.3 Strength and Weakness of the Modelling	34
Table 2.4 An Overview of HOMER	37
Table 2.5 Notations for Goal Model	39
Table 2.6 Overview of Role Model	40
Table 3.1 HOMER for Voting System	49
Table 3.2 Notations for Goal Model	52
Table 3.3 HOMER Mapping to Goal Model	52
Table 3.4 Overview of Role Model	55
Table 3.5 Mapping from HOMER to Role Model	55
Table 3.6 Role Model for Host	55
Table 3.7 Role Model for Secretary	56
Table 3.8 Role Model for Vote Verifier	56
Table 3.9 Role Model for Shareholder	57
Table 3.10 Blockchain-enabled Voting System Scenario Model	65
Table 3.11 The Transformation Guideline	70
Table 4.1 Elicitation Question and Answer	73
Table 4.2 Role Description of Government	81
Table 4.3 Role Description of Banker	81
Table 4.4 Role Description of the Lottery Company Manager	81
Table 4.5 Role Description of the Lottery Intermediary	82
Table 4.6 Role Description of the Lottery Game Player	82

Table 4.7	Scenario Model 1 for Ethereum Blockchain-based Lottery Application	91
Table 4.8	Scenario Model 2 for Ethereum Blockchain-based Lottery Application	92
Table 4.9	Scenario Model 3 for Ethereum Blockchain-based Lottery Application	93
Table 4.10	The Transformation Guidelines for Blockchain-based Lottery Application	98
Table 5.1	Definition of the Blockchain Properties	107
Table 5.2	Requirements Based On The Case Study	111
Table 5.3	Mean Score for UML and AOM	113
Table 5.4	The Blockchain Requirements for the Lottery Case Study	118
Table 5.5	Scoring Criteria for Experiment 3	123
Table 5.6	Comparison of Experiment 2 and Experiment 3 in Percentage	126
Table 5.7	Mean Score for AOM and Extended AOM	126
Table 5.8	The Set of Questions in the Survey	127

LIST OF FIGURES

	Page
Figure 2.1 The ABCDE Process (Marchesi et al., 2019)	23
Figure 2.2 The Activity Diagram of the Methodology (Jurgelaitis et al., 2019)	25
Figure 2.3 <i>e3</i> value model notations (Huemer et al., 2008)	43
Figure 3.1 Extended Agent-Oriented Modelling	47
Figure 3.2 The meeting and voting processes	48
Figure 3.3 An Overview Goal Model for Voting System	54
Figure 3.4 Organisational Model for Voting System	58
Figure 3.5 Domain Model for Voting System	59
Figure 3.6 <i>e3</i> value Model for Traditional Voting System	61
Figure 3.7 <i>e3</i> value Model for Blockchain-enabled Voting System	61
Figure 3.8 The Agent Role Model Mapping	63
Figure 3.9 Knowledge Model for Blockchain-enabled Voting System	64
Figure 3.10 Interaction Model for Blockchain-enabled Voting System	65
Figure 3.11 Behaviour Model for Blockchain-enabled Voting System	68
Figure 4.1 The Goal Model of Govern Lottery Industry	78
Figure 4.2 Goal Model for Manager Manage the Lottery Company	79
Figure 4.3 Goal Model for Winning a Fortune by the Player	80
Figure 4.4 Organisation Model for Lottery Industry	83
Figure 4.5 Domain Model for the Lottery Case Study	84
Figure 4.6 <i>e3</i> Value Model for Traditional Lottery Company	85
Figure 4.7 <i>e3</i> Value Model for Lottery Company	87
Figure 4.8 Role Mapping of the Actors and Agents	88

Figure 4.9	Interaction Model When Player Join Bet in Blockchain-based Lottery Application	90
Figure 4.10	Knowledge Model for Blockchain-based Lottery Application	91
Figure 4.11	Behaviour Model for Player Join a Bet in Blockchain-based Lottery Application	96
Figure 4.12	Behaviour Model for Manager Select a Winner in Blockchain-based Lottery Application	97
Figure 4.13	The Main Page of the Lottery Application	100
Figure 4.14	The User Interface of Rules and Regulations	100
Figure 4.15	Prompt Message When Player Does Not Checks on the Agreement	101
Figure 4.16	When Player Enters the Amount and Checked on the Agreement	101
Figure 4.17	Display Metamask After Player Joins the Bet	102
Figure 4.18	Redirect Player to the Success Page Once Transaction Done	102
Figure 5.1	The Summary of Students' Marks	113
Figure 5.2	Survey Result Analysis	114
Figure 5.3	The Benefit of AOM to Model Blockchain Application Development	120
Figure 5.4	The Survey of AOM Modelling Notations in General	122
Figure 5.5	The Summary of the Final Result	125
Figure 5.6	The Overview of the Platform Independent Modelling Layer Result	126
Figure 5.7	The Response of the Understanding of the Case Study	128
Figure 5.8	The Feedback on the Extended AOM in General	129
Figure 5.9	The Responses for Extended AOM From Different Perspectives	130
Figure 5.10	The Feedback on the $e3$ Value Model	131
Figure 5.11	The Feedback on the Strength of the Extended AOM	132

LIST OF ABBREVIATIONS

ABCDE	Agile Block Chain Development Engineering
AOM	Agent Oriented Modelling
CIM	Computation Independent Modelling
DAPP	Decentralised Application
FCSIT	Faculty of Computer Science and Information Technology
HOMER	Human-Oriented Method for Eliciting Requirement
MDE	Model Driven Engineering
PIM	Platform-Independent Modelling
PSM	Platform-Specific Modelling
UML	Unified Modelling Language

CHAPTER 1

INTRODUCTION

1.1 Introduction

With the rapid growth of innovative technology, blockchain technology has received much attention.

Blockchain has introduced a new decentralised concept to solve synchronised problems in a traditional database (Lin & Liao, 2017). Blockchain technology uses a consensus mechanism to ensure all blockchain nodes agree on the same message, ensures that the newest block has been appropriately added to the chain, guarantees that the message stored by each node is the same, and protects against malicious attacks. A node is created when a user joins the blockchain network. The primary function of a blockchain node is to verify each block of network transactions. A unique identification distinguishes each node from the others. The data will not be lost easily, even if one of the nodes fails or encounters a physical disaster.

The main advantages of blockchain technology are a decentralised ledger, immutable blocks, and strong security (Kim & Laskowski, 2018; Nakamoto, 2008). Blockchain replicates the data over the nodes that do not require any governing authority. Since there is no central governance point for a blockchain network, the blockchain users verify and validate the data as a node. The replication and synchronisation of data across networks help multiple organisations trace and track data.

Imagine if the blockchain network has one million nodes (users). It means there are millions of records in the blockchain network. For security purposes, the malicious actor is

required to alter 51% of the nodes to make changes simultaneously (Lin & Liao, 2017). Hence, such strong security protection provided by blockchain can enhance the prevention of fraud and unauthorised activities.

Blockchain is used in healthcare systems to manage electronic health records (EHRs), validate and track patients' medical history, and trace research methods on fabricating drugs (Kombe, Ally, & Sam, 2018). The patient's data is securely held on blockchain and is accessible and traceable by another hospital/clinic through the healthcare system. When an accident occurs, and a patient is sent to the nearest hospital, the medical staff can easily access the patient's information accurately and instantly. At the same time, blockchain technology will ensure the security of the patient records by tracing who accessed the information and controlling the authorisation to amend the records (Azogu, Norta, Papper, Longo, & Draheim, 2019).

In the financial industry, blockchain has removed the central authority and enhanced trust (Sukheja, Indira, Sharma, & Chirgaiya, 2019). The blockchain is treated as a global ledger to record financial transactions, asset management, insurance claims, and global transactions (Sukheja et al., 2019). With blockchain, the financial industries can trace transactions and supporting documentation effectively and transparently. Meanwhile, the clearing and settlement of transactions can be handled transparently, at low risk and effectively (Knezevic, 2018). Using blockchain will increase trust in any transaction in a secure manner.

The blockchain's decentralised, transparent, and synchronised behaviour in supply chain management leads to transparent and accurate end-to-end tracking. Blockchain does benefit the supply chain management business when two or more organisations (manufacturer, supplier, seller, delivery, and buyer) cooperate. Organisations can digitise

physical assets and generate a decentralised, immutable record of all transactions, allowing for asset tracking from manufacturing to delivery or end-user use. Meanwhile, the status of each physical asset can update and record on the blockchain network in nearly real-time. As a result, traceability and transparency can improve organisational interactions by improving productivity, lowering risks, and being cost-effective. (Aslam, Saleem, Khan, & Kim, 2021; Bumblauskas, Mann, Dugan, & Rittmer, 2019).

Smart contract was introduced in 1994 by Nick Szabo. It is a digital agreement between two people written in computer code (Lallai, Pinna, Marchesi, & Tonelli, 2020). A smart contract is a digital self-enforcing contract with rules and agreements encapsulated in lines of codes in object-oriented programming such as Solidity, Golang, JavaScript, C++, Java, etc. According to the founder, a vending machine is the most appropriate example of a smart contract. Consumers buy and get the item at a vending machine by paying the amount of money without the existence of the middleman (the contract enforcer). In essence, a smart contract allows us to perform a transaction without formal legal documents and the existence of an intermediary (lawyers and judges). Compared to the traditional way of performing a transaction, the smart contract would help achieve the same purpose autonomously, and the transaction is traceable and immutable. The idea behind this smart contract is to increase the level of trust of the users and live forever on the blockchain network.

A smart contract is an agreement in a code that will automatically enforce and make a transaction when the condition is met. A smart contract is invented to eliminate the existence of a middleman (Poels, Kaya, Verdonck, & Gordijn, 2019). Using smart contracts will save documentation, time, and cost. Furthermore, it could solve the trust issue when trading online involving multiple parties like supply chain management. Imagine when a buyer buys an asset or liability. A smart contract replaces a lawyer and automatically

examines the agreements made in the contract. A transaction is performed when the agreement is achieved.

On the other hand, a smart contract can reduce costs and improve efficiency. However, the smart contract does more than replace the contract in the real world. Still, they strengthen themselves in the software world by adding decentralisation, autonomy, and immutability.

Ethereum is a blockchain-based platform founded by Vitalik Buterin in 2013 and released in 2015. It is the most common platform for creating a smart contract in the Solidity programming language. The smart contract can be developed using open-source code tools such as Remix straight from the browser. In addition, Remix supports debugging, testing, and deploying to ease development. The code compilation will return the Application Binary Interface (ABI) in JSON format and bytecode. ABI is a list of functions and arguments in a JSON format to access binary data in the contract. Bytecode is binary data converted from a smart contract that will run and be stored on Ethereum Virtual Machine (EVM) under a contract address. EVM is a runtime environment that runs on every node to execute smart contract bytecode.

Gas is required as an execution fee to perform a transaction and execute a smart contract. Gas is a unit measurement for computational performance to reward the miners; any operation executed on Ethereum requires some gas. The unit for gas in Ethereum is known as Ether. A crypto wallet that holds the Ether can interact and transfer gas to an account to account or account for a smart contract. The most common crypto wallet is MetaMask, Mist, and MyEtherWallet. Ethereum accounts are classified into two types which are Externally Owned Accounts and Contract Accounts. An externally Owned Account is the wallet address controlled by the private key and holds an ether balance to perform a

transaction. Contract Account can only be activated by Externally Owned Account. The contract account has an associative code and contract balance. It is activated when there is a transaction. The contract address can be retrieved when the contract owner deploys the contract to the Ethereum network. Gas is required to deploy a contract; the gas fee is to pay EVM for the executive power; the higher the gas fee, the faster the block can be mined.

Software engineering methodologies have been introduced to aid blockchain application development. However, more study is needed to address standard tools and methodologies for implementing blockchain applications, especially software engineering methodology and framework for designing blockchain applications (Jiang et al., 2022). It has been reported that blockchain developers should focus more on software engineering practices to have a specific and systematic way to analyse and ensure the software quality in the blockchain development process (Chakraborty et al., 2018). In addition, proper software engineering practices could provide crucial actions, guidelines, checklists, design principles, and heuristics for mitigating, tracing, and correcting the source of vulnerabilities and errors (Fahmideh et al., 2022). These practices are essential to guarantee high-quality software that achieves the desired results while staying within budget and time restrictions. Meanwhile, using software engineering practices in a blockchain application to increase sustainability is crucial (Lund et al., 2019).

Although the methodologies have been introduced (Fridgen et al., 2018; Marchesi, Marchesi, & Tonelli, 2018; Ibba, Pinna, & Pani, 2017; Frantz & Nowostawski, 2016), there are still problems pertaining to unsuitable software models in demonstrating the internal architecture of the blockchain, such as gas transaction and decentralisation of data.

Moreover, all the models are generic and do not take into blockchain concepts as first-class entities. As blockchain is a new emerging technology, it can be challenging to

understand the technical concepts and handle unforeseen constraints and uncertainties (Almeida, Albuquerque, & Silva, 2019).

Hence, a gap exists in using, extending, and transforming the current blockchain modelling and practices when developing a blockchain application. This research introduces insight into blockchain application development through extended Agent-Oriented Modelling (AOM). AOM is a methodology for complex socio-technical system development and can become an alternative methodology for blockchain application development. As AOM has not been explored in blockchain application development, to what extent it is useful for blockchain application development is yet to be explored.

This research introduces an extended AOM to provide a systematic approach to developing a blockchain application. With extended AOM, the blockchain application developer can identify the problems, determine the needs of the blockchain, and design the blockchain application.

1.2 Problem Statement

Blockchain technology is a new emerging technology and complicated. Blockchain developers' skills and experience are still immature. Hence, blockchain developers should follow a standard Software Engineering practice throughout development. Bosu, et al. (2019) addressed that current blockchain development lacks modelling tools, and metrics could not better represent blockchain applications. Traditional use case diagrams, activity diagrams and state diagrams are unsuitable for representing blockchain applications.

It still needs clarity on how the application should be performed (Deshpande, Stewart, Lepetit, & Gunashekar, 2017). The immaturity of blockchain technology commits a need for more understanding for developers and stakeholders to implement blockchain on

potential use cases. Fridgen, et al. (2018) mentioned the importance of systematic methodology to help us to understand and create blockchain use cases. Gordijn, Wieringa, Ionita, and Kaya (2019) mentioned that a sustainable blockchain use case is crucial to the business ecosystem. Having a proper methodology is like a project management technique to lead us to plan, design, develop, and analyse a project. Clarity on implementing smart contracts through blockchain may restrict the functionality of the smart contract. Many people still misunderstand smart contracts as electronic contracts converted from traditional ones (Deshpande et al.,2017).

To support blockchain application development, software development methodology is vital to serving as a guide and blueprint. While software methodologies are introduced, there still needs to be more work to elicit blockchain requirements, perform analysis, design, and implement blockchain applications. Blockchain can be challenging to understand technical concepts and handle unforeseen constraints and uncertainties, especially for novices. Hence, there is a gap in developing blockchain applications that lie in understanding how to adapt and apply software methodology to the unique challenges of blockchain technology and developing new methodologies and practices tailored to the requirements of blockchain development to create a sustainable use case.

1.3 Research Objectives

The objectives of this research are:

- i. To compare the suitability of Agent-Oriented Modelling (AOM) and Unified Modelling Language (UML) notations for modelling blockchain applications to identify the notation tailored to the requirements of blockchain development to create a sustainable use case.

- ii. To design an extended methodology for blockchain-based application development based on Agent-Oriented Modelling that is suitable for requirement analysis, design, and model transformation for blockchain application development.
- iii. To evaluate the usability of the extended methodology through case studies and experiments.

1.4 Research Questions

Based on the research objectives, this study aims to answer the following research questions:

- i. Is there a statistical difference in the ability of UML and AOM to capture blockchain requirements in the development of blockchain applications?
- ii. How could the extended AOM provide significant results on requirement analysis, design, and model transformation for blockchain application development?
- iii. What are the benefits of using the extended methodology based on Agent-Oriented Modelling for blockchain application development, and how do these benefits compare to other methodologies?

1.5 Scope of the Research

This research focuses on adopting Agent-Oriented Modelling notation to model blockchain application development for the Ethereum platform. The scope of the research is focused on conducting experiments among novices at University Malaysia Sarawak to evaluate the usability of an extended methodology for blockchain-based application development based on Agent-Oriented Modelling (AOM). The study will involve students

who are new to blockchain technology and have limited experience in software engineering. The experiments will be designed to assess the usability of the extended methodology and determine whether novices can effectively use the methodology to capture blockchain requirements. The scope of the research is limited to evaluating the extended methodology in a controlled experimental setting and does not aim to generalize the findings to other contexts or populations.

1.6 Expected Outcome

An extended AOM is proposed and introduced to aid blockchain developers in identifying and analysing requirements, designing, and implementing a blockchain application. The proposed methodology will help the novices to understand blockchain application development better.

1.7 Thesis Structure

This thesis consists of seven chapters. The summary of each chapter is explained as follows:

Chapter 1 presents the research idea, the background of the studies, the problem statement, identify research questions, the research objectives, the scope, and the expected outcome.

Chapter 2 justifies the research questions stated in Chapter 1 based on the related literature review's findings. Chapter 2 included the software methodologies for blockchain application development, the existing methodologies used by the experts in the field, the limitations of each methodology, and the introduction of agent-oriented modelling.

Chapter 3 gives an overview of the proposed solution — introducing the development methodology and elaborating on the case studies. Additionally, the proposed solution is explained and adapted in the case study.

Chapter 4 explains the lottery case study and demonstrates the extended AOM on the case study. The lesson learnt from the case study is included in this chapter.

Chapter 5 presents the evaluation of UML, AOM, and extended AOM to investigate the modelling technique's usability and comprehensibility in deriving blockchain requirements. Three experiments are conducted with the final year students from software engineering to examine the modelling techniques.

Chapter 6 concludes this thesis by summarising the findings based on the analysis, highlighting the contributions, and establishing recommendations for future work.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

This chapter reviews the software methodologies for blockchain application development and the background knowledge in this research. Section 2.2 presents the software engineering methods for blockchain application development. Section 2.3 shows the analysis of the reviewed methodologies and discusses the challenges when adopting the software methodologies in blockchain application development. Section 2.4 summarizes the agent-oriented modelling, which serves as the background knowledge in this research. The chapter is concluded in Section 2.5.

2.2 Software Methodologies for Blockchain Application Development

Blockchain application development is complex as it has many architectural components, such as smart contracts and wallets (Zeng & Zhang, 2019; Aggarwal & Kumar, 2021). Each actor and component in a blockchain application contribute significantly to the network's security and reliability. The rapid deployment of blockchain technologies results in massive disasters, such as security attacks by owning a poor design and architecture and poor software development practices (Lenarduzzi, Lunesu, Marchesi, & Tonelli, 2018).

Hence, every blockchain development must have a transparent and clear development process and take advantage of the modelling notations (Porru, Pinna, Marchesi, & Tonelli, 2017). With an excellent technical knowledge of blockchain technology, the development team can reach a consensus on blockchain application development (Udokwu, Anyanka, & Norta, 2020).

Several software engineering mechanisms have been adopted for blockchain application development to fill the blockchain development gaps.

They are software development life cycle (SDLC) (Miraz & Ali, 2020), Action Design Research (ADR) Approach (Fridgen, Radszuwill, Rieger, & Urbach, 2018), agile methodology (Ibba, Pinna, & Pani, 2017; Mannaro, Pinna, & Marchesi, 2017; Lenarduzzi et al., 2018; Montes, Larios, Avalos, & Ramírez, 2018; Ibba et al., 2018; Daud et al., 2022), agile block chain DApp Engineering (ABCDE) (Marchesi, Marchesi, & Tonelli, 2018; Marchesi, Marchesi, & Tonelli, 2019; Baralla, Pinna, & Corrias, 2019; Lallai, Pinna, Marchesi, & Tonelli, 2020; Cocco et al., 2021), Model-Driven methodology (Frantz & Nowostawski, 2016; Jurgelaitis et al., 2019; Xu, Weber, & Staples, 2019; Norta & Udokwu, 2019; Udokwu et al., 2020; Abbas, Rashid, Azam, Rasheed, & Anwar, 2021), Entity Relationship Diagram (ERD), Unified Modelling Language (UML), Business Process Modelling Notation (BPMN) (Rocha & Ducasse, 2018a), and e^3 value model (Poels et al., 2019). The summary of the software methodology for blockchain application development is shown in Table 2.1.

Table 2.1: Software Methodologies for Blockchain Application Development

No	Title	Source	Methodology	Modelling	Evaluation Method
1	Blockchain-Enabled Smart Contract-Based Applications: Deficiencies with the Software Development Life Cycle Models (Miraz & Ali, 2020)	Research Gate	Software Development Life Cycle	NIL	NIL
2	A Solution in Search of a Problem: A Method for the Development of Blockchain Use Cases (Fridgen, Radszuwill, et al., 2018).	Research Gate	Action design research h (ADR), Situational method engineering (SME)	NIL	Qualitative analysis
3	CitySense: blockchain-oriented Smart Cities (Ibba et al., 2017)	ACM Digital Library	Agile approach	NIL	Qualitative analysis
4	Crypto-Trading: blockchain-oriented energy market (Mannaro et al., 2017)	IEEE	Agile approach	NIL	Qualitative analysis
5	Blockchain applications for Agile methodologies (Lenarduzzi et al., 2018)	Research Gate	Agile approach	NIL	Qualitative analysis
6	Applying Blockchain to Supply Chain Operations at IBM Implementing Agile Practices in a Smart City Environment. (Montes et al., 2018)	Research Gate	Agile approach	NIL	Qualitative analysis

Table 2.1 continued

7	Initial Coin Offerings and Agile Practices (Ibba et al., 2018)	Research Gate	Agile approach	NIL	Qualitative analysis
8	Sustainable Social Bank Development using Blockchain (Daud et al., 2022)	IJPCC	Agile approach	UML diagram	Qualitative analysis
9	An Agile Software Engineering Method to Design Blockchain Applications (Marchesi et al., 2018)	Research Gate	Agile approach - ABCDE	UML diagram	Qualitative analysis
10	ABCDE –agile blockchain DApp engineering (Marchesi et al., 2019)	Science Direct	Agile approach - ABCDE	UML diagram	Quantitative analysis
11	Ensure Traceability in European Food Supply Chain by using a blockchain System (Baralla et al., 2019)	IEEE	Agile approach - ABCDE	UML Diagram	Qualitative analysis
12	Software Engineering for DApp Smart Contracts managing workers Contracts (Lallai et al., 2020)	Research Gate	Agile approach - ABCDE	UML diagram	Qualitative analysis
13	A Blockchain-Based Traceability System in Agri-Food SME: Case Study of a Traditional Bakery (Cocco et al., 2021)	IEEE	Agile approach - ABCDE	UML diagram	Qualitative analysis

Table 2.1 continued

14	From Institutions to Code: Towards Automated Generation of Smart Contracts (Frantz & Nowostawski, 2016)	Research Gate	Model-Driven Engineering	NIL	Qualitative analysis
15	Modelling principles for blockchain-based implementation of business or scientific processes (Jurgelaitis et al., 2019)	Semantic Scholar	Model-Driven Engineering	UML diagram	NIL
16	Model-Driven Engineering for Blockchain Applications (Xu, Weber, & Staples, 2019)	Springer	Model-Driven Engineering	BPMN	Qualitative analysis
17	A Model-Driven Framework for Security Labs using Blockchain Methodology (Abbas, Rashid, Azam, Rasheed, & Anwar, 2021)	IEEE	Model-Driven Engineering	Meta model	Qualitative analysis
18	A Blockchain System for the Attestation and Authorisation of Digital Assets (Norta & Udokwu, 2019)	Research Gate	Model-Driven Engineering	AOM Goal Model, UML Component diagram, UML Sequence diagram	Quantitative analysis

Table 2.1 continued

19	Evaluation of Approaches for Designing and Developing Decentralised Applications on Blockchain (Udokwu et al., 2020)	Research Gate	Model-Driven Engineering	AOM Goal Model, UML Component diagram, UML Sequence diagram	Quantitative analysis
20	Preliminary Steps Towards Modeling Blockchain Oriented Software (Rocha & Ducasse, 2018a)	IEEE	NIL	UML, ERD, BPMN	Qualitative analysis
21	Early Identification of Potential Distributed Ledger Technology Business Cases Using e^3 value Models (Poels et al., 2019)	Research Gate	Model Driven Engineering	e^3 value model	Qualitative analysis

2.2.1 Software Development Life Cycle

Miraz and Ali (2020) pointed out the limitations of the SDLC models for blockchain development. Due to the characteristics of immutability behaviour, consensus mechanism, and gas cost during the transaction in a smart contract, it limits the SDLC in blockchain application development. The SDLC models (Waterfall Model, V-shape Model, Iteration and Incremental Model, Spiral Model, Big Bang Model, and Agile Model) need to be more suitable for blockchain-based systems, especially in the testing and maintaining blockchain applications. In this case, testing the blockchain application requires a high volume of transactions, making verification and validation work unreachable and impractical. Meanwhile, maintaining the blockchain application is impossible to its immutability behaviour. Hence, the authors urged the formulation of a new SDLC model to resolve the issues in blockchain application development through SDLC.

Haeri et al. (2022) address that System Development Life Cycle (SDLC) tend to cause cost and performance. Such problems might invalidate design decisions, necessitating a complete redesign or implementation change. It can result in schedule and expense overruns, as well as the termination of projects. As a result, there is an urgent need for resource cost and system performance verification and validation and for this to be part of the continuous design process instead of being added late in the development.

2.2.2 Action Design Research and Situational Method Engineering

Fridgen et al. (2018) addressed the need for a technology-driven and systematic approach to design and develop potential blockchain use cases. The researchers apply the action design research (ADR) and situational method engineering (SME) to the blockchain use case development (BUD) method to identify the potential of blockchain and design persuasive use cases. According to Fridgen et al, the BUD can deliver constructive

blockchain knowledge to organisations utilising blockchain. However, the authors mentioned that the limitation of the method has yet to demonstrate its applicability to other emerging technologies, and the future uncertainty of new characteristics of blockchain might affect the BUD method.

The ADR approach consists of four steps: "problem formulation", "building, intervention, and evaluation", "reflection and learning", and "formalisation of learning". SME is integrated into ADR for method creation, which involves three steps: specifying the requirements of the method, selecting appropriate method chunks, and assembling the chosen method chunks.

The problem formulation defines an issue faced by the researchers. Then, the methods or constructs are evaluated throughout the development process. Next, learn from the lesson through reflection. The final step is to generalise the learning outcome.

The method was tested in four workshops across diverse industries (automotive, banking, construction, and insurance) to ensure the applicability of the results. The authors made significant contributions by providing practitioners with a structured and theoretically robust approach for utilising blockchain technology and generating valuable insights in the areas of digitalisation, innovation management, and emerging technologies. Nevertheless, the method still needs to demonstrate its effectiveness and relevance in the context of other emerging technologies.

2.2.3 General Agile Approach

Lenarduzzi, Lunesu, Marchesi, and Tonelli (2018) proposed an agile approach to blockchain application development. The agile approach is a flexible, iterative, and adaptive

methodology that involves steps to gather requirements, development, testing, gathering feedback from users and enhancement.

Ibba et al. (2017) applied agile scrum methodology in a blockchain application, “CitySense System”, because of its flexibility, adaptability, and iterative capabilities. Scrum is a methodology that involves daily meetings with the team and stakeholders. Scrum is carried out in brief sessions where changes are observed and discussed for future development and the time required to complete each task. Scrum and agile methodologies are used to ensure that a product's functionality is ensured and that it can be showcased quickly. Scrum allows developers to finish feature development and testing in a short iteration. Meanwhile, scrum focuses on the process rather than specific software development approaches (Marchesi et al, 2019).

Ibba et al. (2018) highlight that agile methodology suit blockchain technology start-ups. The author pointed out that the methodologies are flexible and able to manage uncertainty. In blockchain development, an agile roadmap was designed as a milestone and timeline for the blockchain project. A roadmap is a valuable tool for fostering progress accountability and managing clients’ demands. First, the roadmap must cover project objectives. Second, the development team and the stakeholders must communicate to gather feedback for possible improvements. Meanwhile, the roadmap must be agile to accept changes based on feedback. Finally, the immutability blockchain issue must be considered when dealing with changes. Hence, all projects must have strict tests, audits, and verification before deployment.

The literature review of the Agile approach utilized a qualitative research method, explicitly employing case studies for data collection and analysis. The case studies or examples adopted in the literature are CitySense (Ibba et al., 2017), Crypto-Trading project

(Mannaro et al., 2017), case study for product owner (Mannaro et al., 2017), Supply chain operation (Ibba et al., 2018), Initial coin offering (Ibba et al., 2018), and Sustainable Social Bank (Daud et al., 2022).

2.2.4 Agile Block Chain Development Engineering (ABCDE)

Marchesi et al. (2019) propose an Agile Blockchain Agile Development Engineering (ABCDE) for blockchain application development. The ABCDE was introduced due to the complexity of the smart contract. A smart contract introduced new concepts such as the contract's address to allow transaction performance, the gas required to execute the contract, and digital money transfer between two parties (Marchesi et al., 2019).

The ABCDE methodology is adopted in DApp projects such as Food Supply Chain (Baralla et al., 2019), worker contracts management (Lallai et al., 2020), and supply chain management (Cocco et al., 2021). This methodology's advantages help prescribe the actors and user stories from the beginning, simplifying the system and smart contract configurations. Besides, this action lowers the chances of failure since the modelling notations highlighted all the essential actors, requirements, and all connections.

As shown in Figure 2.1, the description of the Agile Block Chain Development Engineering processes is as follows (Lallai et al., 2020; Marchesi et al., 2018, 2019):

- i. **Identify and summarize the goal:** Summarize the system goal into 10-30 words and post the goals to all developers.
- ii. **Identify goal:** State all the related goals and actors in a system.
- iii. **State user stories:** List down all the requirements in terms of user stories. At this stage, the developer can map the user stories and the actors using a UML use case diagram. The use case diagram has captured the user requirements for the entire

DApp system in development. The remaining steps will be taken if the developer adopts blockchain technology.

iv. **The development process is split into two subsystems:**

- a. The smart contract application
- b. The app system acts as a platform to handle events that may occur when a smart contract function is triggered and interactions with the blockchain and users.

v. **Design of the blockchain component:** The following steps are performed iteratively to ensure the completeness of the smart contract.

- a. Redefine the required actor and user stories and target the actor that connects with the smart contract. It is required to specify any existence of the external smart contract of the DApp development in this step.
- b. Specify the functionalities and components of the smart contract, such as the libraries, architecture, and interface. The UML Class diagram is defined the smart contract components.
- c. Specify the message flows and transactions among the smart contract, external smart contract, and the App system. The UML sequence diagram and UML statecharts are used in this step. The UML sequence diagram models the interaction flow of the system, and UML statecharts capture the state change of the smart contract.
- d. Specify the data structure, application binary interface (ABI), and events.
- e. Specify the functionality and the modifiers.
- f. Specify the steps of testing and security assessments.

vi. **Code and test smart contract system.**

- a. The smart contract system is developed and tested in an iterative behaviour based on an agile approach.
 - b. Execute security assessment.
 - c. Write unit tests and acceptance tests by using Truffle.
- vii. **Construct the App system:** The app system is similar to the standard web application but adds blockchain components.
 - a. Reformulate the required actors and user stories by adding new actors, such as a smart contract.
 - b. Design the architecture of the system, which includes the blockchain component.
 - c. Design the user interface for the system.
 - d. Design the complete module and the system flow, including the interaction between human actors and system actors.
 - e. Execute a test on security for the app system.
- viii. **Code and test the App system.** The App system is implemented and tested by using an agile approach iteratively.
- ix. Perform integration, test, and deploy the DApp system.

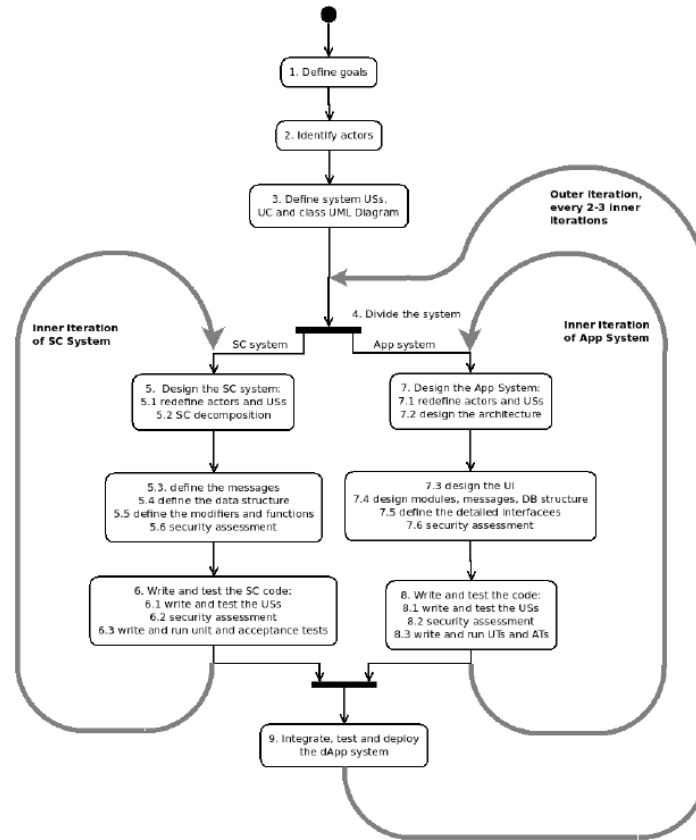


Figure 2.1: The ABCDE Process (Marchesi et al., 2019)

In the ABCDE methodology, UML diagrams such as use case diagrams, class diagrams, sequence diagrams, statecharts diagrams are used to capture the interaction and connection of the actors. Use case diagrams is aimed to map the connection of the actors and user stories before the code development. The class diagram constructs the smart contract's structure and relationship, such as the private and public functions, modifiers, and libraries. The sequence diagram illustrates the exchange of information among the actors in a sequenced manner. Lastly, the statecharts diagram records the state of the smart contract, the app system, and the transaction status.

ABCDE approach has been evaluated through qualitative analysis. There are four case studies, and one survey has been conducted. The case studies are the voting system (Marchesi et al., 2018), food supply chain (Baralla et al., 2019), Dapp for managing

temporary employment (Lallai et al., 2020), and Blockchain-Based Traceability System in Agri-Food for traditional bakery (Cocco et al., 2021).

According to Marchesi et al. (2019), a survey was conducted among 14 developers and graduate students who utilized the ABCDE method in at least one DApp project. The results showed that overall satisfaction with the method was high, and it was perceived as easy to use. The method was found to be particularly beneficial in the analysis and design phases but received comparatively lower appreciation in gas optimization and final integration, though still above average.

2.2.5 Model-Driven Engineering (MDE)

Model-Driven Engineering (MDE) is a technique to model a software system according to different abstraction levels or viewpoints (Jurgelaitis et al., 2019). With MDE, automated code generation can be derived through model transformation. According to Xu, Weber, and Staples (2019), the advantages of MDE for blockchain application development are to avoid any errors or bugs during the coding periods. Meanwhile, the models do not cover a specific platform, allowing easy migration from one blockchain development platform to the others. Furthermore, a model easy to track and verify the correctness of the system and strengthen the confidence of the business parties in the system.

Jurgelaitis, Butkienė, Vaičiukynas, Drungilas, and Čėponienė (2019) proposed model-driven architecture on blockchain development. To model a blockchain application, the methodology covers Computation Independent Modelling (CIM), Platform Independent Modelling (PIM), and Platform Specific Modelling (PSM), as shown in Figure 2.2. In work, the UML diagram is applied to represent a different aspect of the system. CIM focuses on business or domain modelling. It involves a process to identify blockchain applicability for the business case. The models produced at this level use UML notations such as the

participation interaction model, vocabulary, and business case process model. Then, a series of questionnaires will help identify the appropriateness of blockchain as a use case solution. The modeller proceeds on blockchain design through PIM if the blockchain is suitable based on the CIM modelling and questionnaire. At the PIM level, the models derived are platform independent. Next, the developer will define and apply the transformation rules to the blockchain implementation. Finally, a blockchain artefact is defined for blockchain technology development.

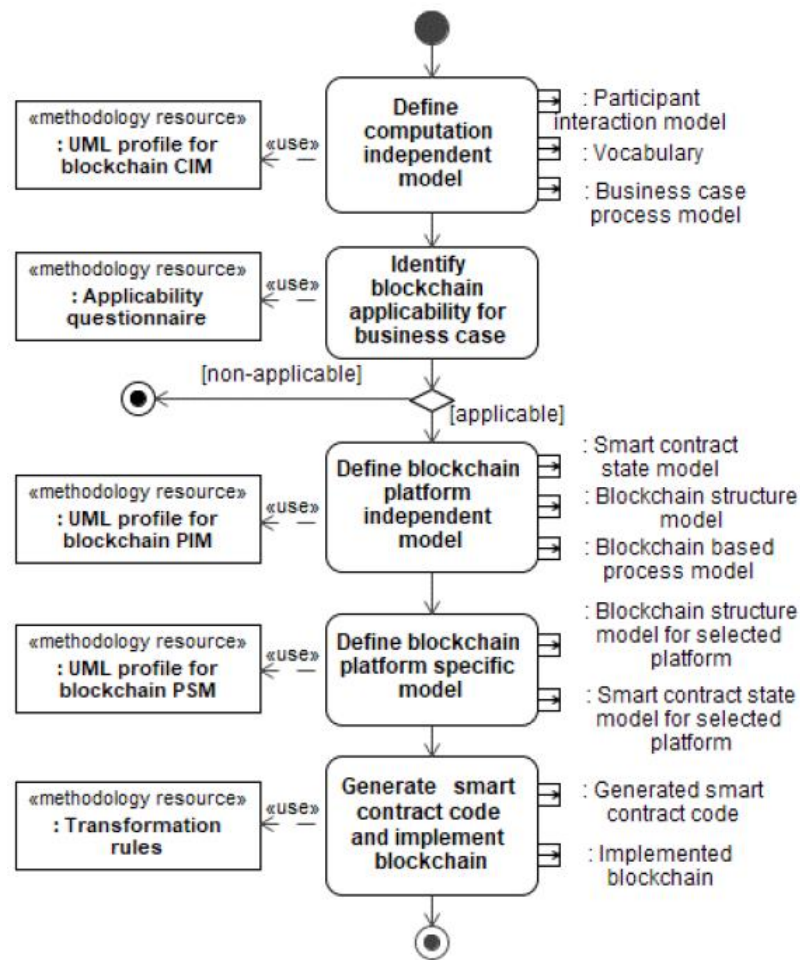


Figure 2.2: The Activity Diagram of the Methodology (Jurgelaitis et al., 2019)

Using a supply chain scenario, Xu, Weber, and Staples (2019) discussed how an architectural decision affects quality, such as the cost and durability of the blockchain application. The system is developed with the Ethereum blockchain. Creating a contract

involves fixed costs and variable costs. The fixed cost is the amount of the transaction and the cost for the contract address. Variable cost is influenced by the architectural design of the contract, such as payload data and costs to execute functions defined in the contract. Along these lines, the model will influence the contract's expense through the contract construction design.

A process model is developed and translated into a smart contract. Then, the method uses the blockchain computing system to manage business processes. Next, the concept of the trigger will be used to connect smart contracts with an enterprise system and the external world. A trigger will be converted to API calls, which bridge blockchain and business processes to the smart.

Frantz (2016) introduced a semi-automation contract generation tool which translates human-readable context into a smart contract. Modelling separate entities in terms of collective adaptive and self-organising processes is crucial. This work makes few assumptions about the interacting entities (e.g., agents, people, contracts, and services), except that they must have access to mutual smart contracts to use the proposed solution in the field of distributed systems in general and collective adaptive systems. This methodology comes with two steps. First, the authors express smart contracts based on the Grammar of the Institution (Crawford & Ostrom, 1995) concept, ADICO. ADICO stands for **A**tttributes, **D**eontic, **A**im, **C**onditions, and **O**r else. The developer translates each requirement into ADICO format. For example, in supply chain management, Driver (A) must (D) deliver (I) the parcel before 5 pm (C). Second, these statements map into a smart contract. This method has been evaluated by using case studies of a voting system and escrow service.

Abbas et al. (2021) resented an MDE-based technique to automatically translate the blockchain metamodels into coding. The technique covers transformation rules as well as

translation tools. The proposed metamodel represents the blockchain concepts such as Blocks, validators, networks, information, information sender node, and information receiver node. A tool known as Siruis's is developed to create and visualise simple and complicated blockchain-based situations in a user-friendly manner. Then the model will translate to code based on a transformation rule that is written in the Acceleo language. This technique increases the productivity of blockchain applications. The author verified this method via a case study in a security-based organisation where several security laboratories are engaged in various projects. The findings demonstrate that the framework is dependable for use and can be expanded to quickly automate and develop security labs' Blockchain-based applications.

Rocha and Ducasse (2018) illustrated the three complementary approaches, which are UML, ERD and BPMN, to modelling, utilising established software engineering models, and applying them to a fidelity point program. Each approach emphasises a specific aspect: data-driven (ER model), structure-driven (class diagram from UML), and process-driven (BPMN notation). Each approach has strengths and weaknesses; a specialised notation is necessary for effective blockchain design.

2.2.6 Agent Oriented Modelling

Norta and Udokwu (2019) pointed out the limitation of current software methodologies in blockchain application development. The limitations are the usability and complexity of blockchain technology understanding by project stakeholders, transaction cost scalability, and security flaws in smart contracts. The author introduced a new framework to overcome the limitation. The framework covers the AOM goal model, UML component diagram and UML sequence diagram. The details of the model aid in reducing and enhancing

the operations of the smart contract. In other words, it could improve scalability and reduce transaction costs as the more operation, the more gas to consume.

First, the framework models the initial requirements by using AOM Goal Model. The goal model represents the main goal and sub-functional in a hierarchy structure. Each goal may be associated with agent/actor and quality goals. Second, the UML component diagram is adopted to illustrate the project architecture and demonstrate the relationship between the system components, related roles, and attributes. This diagram facilitates communication among the stakeholders and developers on the software requirements. Lastly, the UML sequence diagram is adopted to capture the activity and the information the actors or components exchanged.

Norta and Udokwu (2019) conducted two experiments with 10 nodes to assess the performance of the developed system components and ascertain if they are operating at the expected optimal level. In the initial experiment, the throughput and latency of the client components were measured to assess the system's overall performance. In the second experiment, the rate of transaction additions in the blockchain was measured at various slot times to evaluate the performance of block creation. The performance evaluation results indicate that the system's client components possess scalability and can effectively manage increased transaction loads. Furthermore, the evaluated client components exhibit excellent performance and operate optimally. The evaluation of the block-creation component highlights its performance dependency on slot duration.

Secondly, Udokwu et al. (2020) evaluate the framework's usefulness through domain experts in a webinar event. The webinar had 15 participants, including researchers, software engineers, data and business analysts, project managers, and quality-assurance analysts. The DAOM framework underwent evaluation for its semantic and pragmatic qualities and the

effectiveness of its support tool for DApp design and development. The semantic evaluation indicated that the DAOM framework provided an accurate and realistic representation of DApp design and development steps. In contrast, the pragmatic evaluation showed it was instrumental in producing quality DApp designs in organisational settings. The effectiveness of the support tool was assessed by comparing it with a free-hand sketch tool. The results revealed that the DAOM tool required less modelling effort and offered more excellent usability in creating DAOM-diagram models.

2.2.7 e^3 Value Modelling

Poels, Kaya, Verdonck, and Gordijn (2019) utilised the e^3 value model as a framework to evaluate the potential impact of blockchain in several domains, such as energy trading in the decentralised electricity market and online image rights trading platforms.

The author stated that the requirements to develop a sustainable blockchain use case are removing the middleman, sharing the data among the parties, and immutable data. It must achieve at least two requirements to create a sustainable blockchain application.

The author acknowledged the limitation of the approach as identifying potential blockchain business cases still at an early stage is done visually, which enables a rapid and rough evaluation. However, the e^3 value approach allows for the quantification of various model elements such as market segment cardinalities, occurrences of value transactions, valuation of value objects, and adding variable, fixed, and investment costs to value activities, allowing for a net cash flow analysis to evaluate the viability of a business ecosystem.

2.3 Discussion

From the review, blockchain application development is still a challenging process. To start a blockchain project, the developers must be ready with the knowledge and skill for cryptography, networking, security, and distributed computing (Bosu, Iqbal, Shahriyar, & Chakraborty, 2019). Before deployment, testing must be robust and rigorous for the developers to verify the accuracy and quality of the products. However, there are still limited tools or frameworks against the correctness requirements in blockchain application development (Bosu et al., 2019). Hence, blockchain developers should follow a standard software engineering practice throughout development.

On the other hand, there needs to be more documentation of the entire lifecycle of blockchain application development, which is still in its early stages (Deshpande et al., 2017). As it is a new technology, awareness of blockchain knowledge is needed. Also, the blockchain developer must understand the professional roles to design a well-fit blockchain solution project for the stakeholders (Porru et al., 2017).

Action design research and situational method engineering are approaches to developing potential blockchain use cases. It emphasises the creation of practical outcomes in a practical setting, making it highly valuable for addressing practical challenges. However, it needs a standardised or widely adopted methodology, which can pose challenges in comparing and replicating studies or ensuring consistency across different projects or organisations.

SDLC provides a systematic, structured, and easy comprehend approach to software development in blockchain projects, ensuring that all necessary steps are planned and organised. However, it is challenging to implement in the testing and maintenance phase due to the immutable nature of blockchain.

Agile is the best way to define the roles at the beginning steps. Nevertheless, the blockchain knowledge and complexity will need to depend on the modelling notation used by the developers. Even though the strength of agile is to achieve high expectations and satisfaction from the clients, agile will spend more cost on testing. Each test will consume gas or electricity to achieve blockchain consensus to test the correctness of the smart contract.

ABCDE extends the Agile approach to focus on blockchain application development. However, the methodology is provided for smart contract application development on the Ethereum Blockchain platform. Nevertheless, the iteration manner in ABCDE will still affect the cost consumption while implementing and testing.

MDE provides a massive advantage to code generation tools. These code generation tools depend on the modelling notation and translate it into code. In MDE, the modelling notation help to express the knowledge and details of the blockchain. The models are constructed in detail, and it will enhance the correctness of the system. With this, the operation and complexity will be reduced. Furthermore, as a model-based testing technique, the model can assist in developing test cases. (Hametner et al., 2010). As software components in automation systems become more sophisticated, a systematic testing approach may ensure efficient and effective testing in the event of changes.

Although various methodologies have been introduced, it still needs more clarity on how the application should be performed (Deshpande et al., 2017). The immaturity of blockchain technology commits a need for more understanding for developers and stakeholders to implement blockchain on potential use cases. Fridgen et al. (2018) mentioned the importance of systematic methodology to help us to understand and create blockchain use cases.

A proper methodology is like a project management technique to lead us to plan, design, develop, and analyse a project. Clarity on implementing smart contracts through blockchain may restrict the functionality of the smart contract. Many still misunderstand smart contracts as electronic contracts converted from traditional ones (Deshpande et al., 2017).

Table 2.2: Strength and Weakness of Existing Methodology

Methodology	Strength	Weakness
SDLC	A comprehensive approach included requirement analysis, design, development, and testing.	Highly impractical in the testing and maintenance phase due to the immutability of the blockchain.
Action Design Research and Situational Method Engineering	Collaboration and participation of stakeholders throughout the research and design process result in a more inclusive and participatory approach that considers various viewpoints and ideas.	It needs a standardised or widely recognised methodology, making it difficult to compare and reproduce studies or ensure consistency across different projects or organisations.
Agile	It provides for changes in project needs and flexibility. The iterative approach encourages continual improvement by allowing for feedback and enhancing the final result.	Spend more cost on testing to ensure customer satisfaction. It may be useful primarily for small, complex blockchain projects requiring extensive planning and coordination.

Table 2.2 continued

MDE	It enables the creation of exact and consistent models that can be immediately turned into code, guaranteeing that the blockchain application is designed and implemented correctly and without errors.	MDE's high learning curve necessitates extensive information and training to familiarise with the methodology and modelling.
-----	---	--

Rocha and Ducasse (2018) mentioned that each modelling has advantages and disadvantages. UML, ERD, and BPMN cannot sufficiently capture the specification behaviour of blockchain applications, especially the direct connection between the blockchain and the application. Thus, the modelling notation must be further explored to handle the complexity. Bosu et al. (2019) addressed that blockchain development needs modelling techniques and tools. Moreover, traditional use case diagrams, activity diagrams, and state diagrams are unsuitable for representing blockchain applications. On the other hand, blockchain programmers need to be clearer on role definition, whereas smart contracts need industry-specific knowledge such as healthcare, law, or accounting.

To compare AOM to UML, AOM is known for flexibility and adaptability to model complex, dynamic systems with autonomous agents. AOM provides a rich set of modelling constructs for capturing agents' autonomy, proactivity, and social aspects. On the other hand, UML provides a more rigid and standardized set of modelling notations, which may need to be more flexible and adaptable for modelling complex, dynamic systems with autonomous agents. Poels et al. (2019) recognized that the approach has a drawback. It involves visually

identifying potential blockchain business cases that are still in the early stages, allowing for a quick and rough assessment.

Table 2.3: Strength and Weakness of the Modelling

Modelling	Strength	Weakness
UML	UML promotes the use of modular and reusable components, which can help improve the maintainability and scalability of blockchain systems.	UML may only capture a few of blockchain technology's unique features and concepts.
ERD	ERD can aid in designing the underlying database architecture for a blockchain system. It helps to identify and define the tables, attributes, and relationships that will store the data in the blockchain.	ERD may only partially capture some of blockchain technology's unique features and complexities.
BPMN	BPMN offers a standardised and straightforward method for modelling and visualising business processes.	BPMN may not model complex data and the internal structure of blockchain technology.

Table 2.3 continued

AOM	AOM enables the development of complex, dynamic systems with autonomous agents capable of exhibiting emergent behaviour. AOM can aid in distributed decision-making among agents in a blockchain system.	AOM may only capture a portion of the unique characteristics and concepts of blockchain technology.
e^3 Value Model	e^3 value model uses visual representations, such as value diagrams, to depict the value creation and exchange processes in a blockchain ecosystem.	e^3 value model focuses on the value creation and exchange process rather than the blockchain technology's internal structure.

Among the various methodologies, model-driven engineering is promising. Software models will keep and present the concepts simpler. According to Ferre (2010), software modelling provides goals and advantages in term of communication, management, and product implementation. Models can be strong evidence to show the differences between blockchain and traditional applications to convince the stakeholders to adopt blockchain technology. The methodology will assist the blockchain development teams to clearly understand the project's process, including maintaining the project's vision, the requirement request by the client and stakeholders, and the roles involved in the application. Likewise, models will minimize unnecessary mistakes during the development as it shows the logic should be implemented and ease testing and debugging.

In line with the MDA approach, can AOM become an alternative methodology for blockchain application development? Based on the review (Miller, Lu, Sterling, Beydoun, & Taveter, 2014), AOM can serve as a methodology for a non-technical person. It is MDA compliant and designed for complex social-technical system modelling. With AOM, it can easily model the details of the blockchain application and the environment. The background knowledge of AOM is described in the following section.

2.4 Introduction of Agent Oriented Modelling and e^3 value model

The AOM was aimed to benefit technical and non-technical clients in a project. Non-technical clients are the stakeholders, while technical clients are the project manager, front-end developer, back-end developer, and tester. It provides an idea to deal with a complex system to help blockchain developers easily understand and model the application. It is compliant with MDA.

The e^3 value methodology describes the exchange of value (services, products, and money) between actors (suppliers and customers). The primary objective is to pinpoint and examine the process of generating, exchanging, and consuming value among various actors within a network (Huemmer, Schmidt, Werthner, & Zapletal, 2008). In other words, it adopts an economic value viewpoint and visually represents the parties involved in the exchange process and what they are trading.

2.4.1 Elicit Requirements

The methodology starts with gathering information and requirement by using a series of questions called the Human Oriented Method for Eliciting Requirements (HOMER). The questions are designed to be more approachable to non-technical clients (without a blockchain background). They are straightforward and emphasize the “*WHY*” and “*HOW*”

questions to identify their needs, problem, and root cause. This technique aims to lead the stakeholders to inspire and expand their ideas while answering the questions through interviews, self-examination, and group meetings (Miller et al., 2014). Table 2.4 shows the overview of HOMER.

Table 2.4: An Overview of HOMER

Question	Motivation	Model
1 If you are hiring more than one staff, what position do you need to fill?	To define the required role for a system	Goal model
2 Based on the role defined in Q1:		
a. What is the purpose of this role, and what is the solution the role can solve?	This is aimed to specify the primary goal of the system	Goal model
b. What tasks will commonly be required?	Determine the sub-goal to achieve the main goal	Goal model
(i) Is this task performed by a human agent, software agent, or an external system?	This is to assign tasks and set boundaries to an agent	Goal model Role model
c. For each task stated,		
(i) What subtasks make up this task?	To break the main or sub-goal into smaller chunks	Goal model
(ii) What are constraints are there for this task?	To capture the agent behaviour while performing tasks	Role model
(iii) How the role achieves the task?	Illustrate each step of the agent to achieve the main/subgoal	Scenario model
(iv) Does the role depend on another role to achieve its goal?	State the relationship of the agents within the system	Organisational model

Table 2.4 continued

d. For each role,		
(i) Which other roles in the task does this role rely upon?	Show which agent depends on another to accomplish a goal	Organisational model
(ii) Which other roles in this task relies upon this role?		Interaction model
3. What codes of behaviour are required by roles?		
(a) What codes of behaviour must be observed by all of the roles?	To show the behaviour of the other agent.	Behaviour model
(b) Is there any specific code of behaviour for a particular role?	To show the behaviour of the agent.	Behaviour model
(c) Are there any other rules that must adhere to this study?	To elicit any new limitations that roles must work under.	Behaviour model

2.4.2 Modelling

Agent-Oriented Modelling consists of a goal model, role model, organisation model, domain model, knowledge model, scenario model, interaction model, and behaviour model. The AOM complies with MDA, which introduces three viewpoints during the modelling process. The viewpoints are computation-independent modelling (CIM) and platform-independent modelling (PIM).

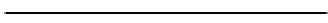
2.4.2.1 Computation Independent Modelling (CIM)

Computation Independent Modelling conveys requirements and demands at a high degree of abstraction without mentioning implementation details.

Goal Model

The goal model is used to illustrate the objective and motivation of the project. The goal model involves a hierarchy structure's main objective and sub-objective. Besides, the goal model displays the quality goal of the system. Goal model aims to maintain the project's vision and mission throughout the development and achieve consensus on the knowledge, problem, and solution between technical and non-technical clients. The goal model is composed of five notations, as shown in Table 2.5 below:

Table 2.5: Notations for Goal Model

Notation	Meaning
	Goal/ subgoal
	Quality goal
	Agent
	Relationship with a quality goal
	Relationship with actor

Role model

A role model outlines the position that exists in the organisation. Roles are defined by three main aspects: responsibilities, permissions, and protocols. The functionality of the agents is derived from their responsibilities. Thus, responsibility can be classified as liveness and safety (Waller, Johnston, & Milton, 2008). Table 2.6 shows the overview of details needed for a role model.

Table 2.6: Overview of Role Model

Role Name	<i>Name of the role</i>
Description	<i>A brief description of the role</i>
Protocols	<i>The protocol that role to be performed</i>
Permission	<i>Authority or the ability of the role</i>
Responsibilities	<i>Functionality or behaviour of the role</i>
Constraint	<i>The condition that will be encounter when accomplishing a goal</i>

Organisational Model

The organisation model represents the relationship of the actors within an organisation. Zambonelli (2011) discovered that relationships between humans and organisations are categorised using control, peer, benevolence, dependency, and ownership. Control will define the role's authority within a system; peer states the equal authority of the roles; benevolence describes the interest shared among the roles; dependency specifies how the roles depend on each other to achieve a goal; ownership determines the delimit organisational limits.

Domain Model

The domain model treats the shared knowledge as domain entities and displays the agent's relationship and the domain entities during any circumstances.

2.4.2.2 Platform independent modelling (PIM)

Platform-independent modelling is the design model which describes the systems' behaviour in the context of stored data and executable algorithms. This phase consists of a knowledge model, interaction model, scenario model, and behaviour model.

Knowledge Model

The knowledge model shows the knowledge or information of the agent required to perform their behaviour. Knowledge models are an ontology that provides a knowledge framework for the problem domain.

Interaction Model

The interaction model defines the protocol between the agents: the purpose, initiator, responder, inputs, outputs, and processing. It shows how the potential information exchange or request in agent interaction patterns is in sequential order. From the interaction model, a stakeholder can understand the agents' interaction and message flow.

Scenario Model

This model contains the steps and strategy of each agent to accomplish a goal in sequential order. The scenario model followed a schema that would display the goal, the initiator, the event to be a trigger, and a list of steps. Each step will be drafted based on the environment the agent will encounter and the agent's response to the environment.

Behaviour Model

The behaviour Model explains an agent's behaviour in an execution cycle (Sterling & Taveter, 2009). An agent's behaviour is defined based on perceptions and knowledge of rules. A start event will trigger a rule when the condition is true and update the agent's mental state. It then sends a message or acts. The agent may perceive the message another agent has sent or a physical action taken by another agent.

e^3 value model

Huemer et al. (2008) stated that actors refer to entities involved in exchanging value and are viewed as autonomous economic units to maximize profits (for enterprises) or enhance their economic well-being (for end-consumers) through the performance of value-generating activities. These activities add value to the business network and are specifically attributed to the corresponding actors. Through these value activities, actors exchange value objects that benefit one or more actors within the network. These objects may include tangible goods, such as physical products or money, and intangible assets, such as services, skills, or expertise.

In the e^3 value model, actors use value ports to indicate their desire to provide or request value objects. The purpose of value ports is to focus solely on external connections between actors and other components rather than on internal processes. A value exchange occurs when two value ports are linked, depicting one or more possible exchanges of valuable objects. Values offered to or requested from the environment are represented by value offerings, which may consist of several value objects. The concept of value offerings allows for mapping "object bundling" when items of value can only be requested or provided together. Actors may have multiple value interfaces grouping individual value ports, and the exchange of values is atomic, meaning value objects are exchanged through the value interface on all its value ports, each exchanging precisely one value object or none.

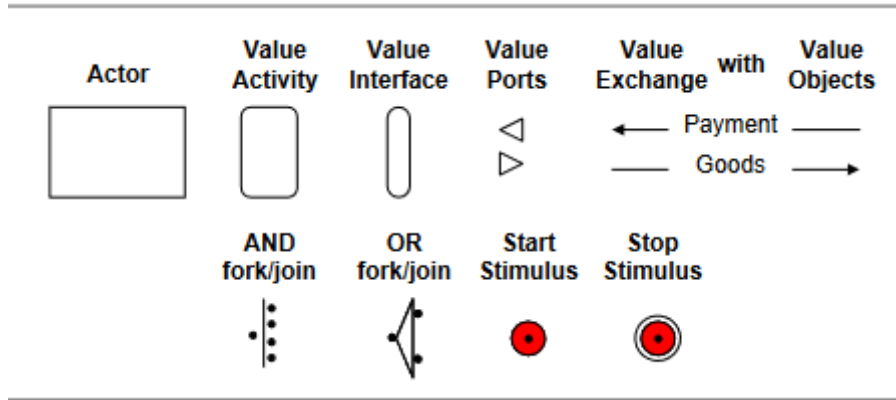


Figure 2.3: e^3 value model notations (Huemer et al., 2008)

2.5 Chapter Summary

From the review, software methodology for blockchain applications is important for blockchain adoption and sustainability. Although various methodologies have been introduced, they lack modelling notation (Rocha & Ducasse, 2018). In line with the advantages of the MDE approach in blockchain application development, it is worth investigating to what extent AOM can serve as the alternative methodology for blockchain application development. The extended AOM for blockchain application development is presented in Chapter 3 by using a voting system case study. The voting scenario and characteristics are referred from the manual voting system (Paatey & Ofori-Dwumfuo, 2011). The following chapter demonstrates how the extended AOM creates a blockchain-based voting system that is derived from a manual voting system.

CHAPTER 3

PROPOSED EXTENDED METHODOLOGY

3.1 Overview

This chapter presents an overview of the proposed methodology for blockchain application development. Section 3.2 presents the overview of the research methodology. Section 3.3 illustrates the rationale for extending the AOM. Section 3.4 illustrate the methodology of the extended AOM through a walk-through example of a blockchain-enabled voting system. Section 3.5 provides the conclusion for the chapter.

3.2 Research Methodology

The research methodology discussed in this information involves four steps. In the first step, investigate the relevant issue topic within the research areas. This step involves studying blockchain technology's state of the art, conducting a literature review of existing blockchain application development approaches, and defining the problem statement and research objectives. A pilot test was conducted among novices to compare the result of UML and AOM.

Step two propose a methodology by extending the AOM (Agent-Oriented Modeling) with the e^3 value model and demonstrating the extended AOM using step-by-step instruction with a voting system case study.

In step three, validate the proposed methodology by using a lottery system as a case study, implementing the modelling into a blockchain application, and comparing the modelling with existing methodologies.

In step four, the extended AOM is evaluated by recruiting 140 students from FCSIT, UNIMAS to participate in experiment 2 and experiment 3. The students participated in a talk by Dr Alexander Norta, a workshop to understand the extended AOM. They were required to model the blockchain-based lottery system using AOM and extended AOM. Students were graded based on their models, answered a survey, and the results of AOM and extended AOM were analyzed and compared to discuss the findings.

Overall, this methodology involves investigating the research area, proposing a methodology, validating it, and evaluating the methodology to gain insights into its effectiveness.

3.3 Rationale in Extending the AOM

Based on the review in Chapter 2, blockchain application development require an alternative methodology. A pilot study was conducted to test the usability of the UML and AOM, and the analysis is further explained in Chapter 5. The results show that the AOM must be extended to trigger the potential blockchain properties, especially among novices.

3.4 Extended Agent-Oriented Methodology

Figure 3.1 shows the proposed methodology for blockchain application development through extended AOM. The extended AOM consists of five phases before implementation. The first phase is requirement elicitation through the Human Oriented Method for Eliciting Requirements (HOMER). Then is followed by computation-independent modelling through goal modelling, role modelling, organisation modelling, and domain modelling. In this case, the elicited answers from HOMER are modelled through the goal model, role mode, organisation model, and domain model. The first phase and second phases are known as problem understanding. Upon the problem and domain understanding, the modeller involves

early identification of blockchain use cases through e^3 value modelling and agent role mapping. Next is followed by platform-independent modelling to design a blockchain-enabling application. Finally, a platform-specific modelling layer is designed. It then focuses on transforming the design model into a blockchain construct.

In the following section, we elaborate on the proposed methodology based on a walkthrough example of a voting system that aims to solve transparency issues using blockchain. The case study illustrates a voting system process in a general meeting starting from an invitation sent by the secretary to shareholders to attend the meeting. During the meeting, eligible shareholders must be able to vote using a paper ballot initiated by the meeting host. A verifier will evaluate and verify the paper ballot before the vote is tabulated. The verifier is responsible for tabulating the paper ballot and announcing the result. The whole process of the meeting is documented and monitored by the secretary, as shown in Figure 3.2.

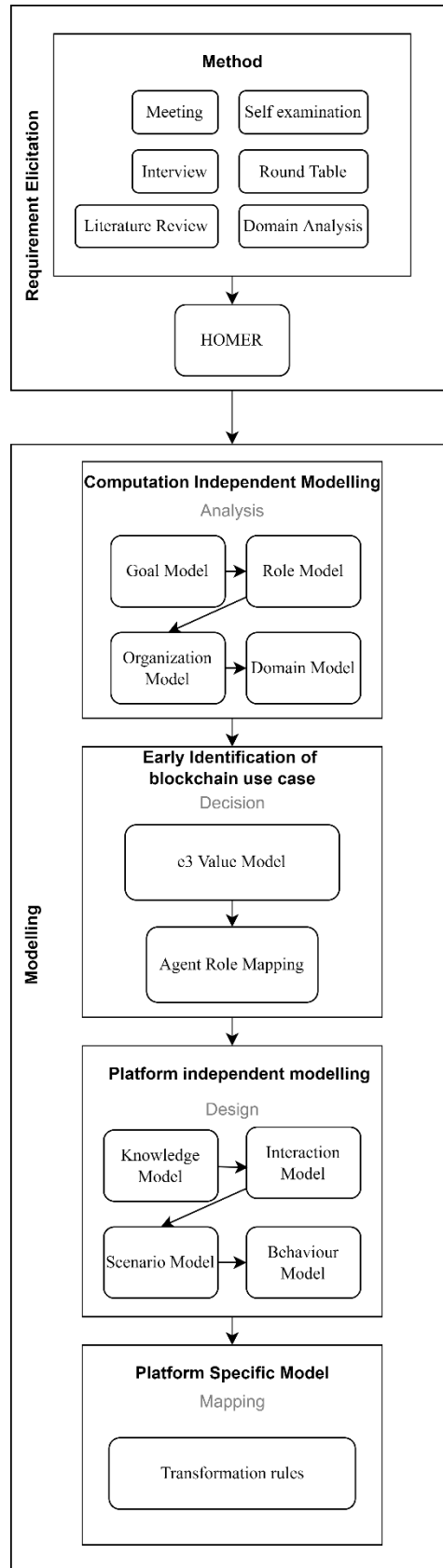


Figure 3.1: Extended Agent-Oriented Modelling

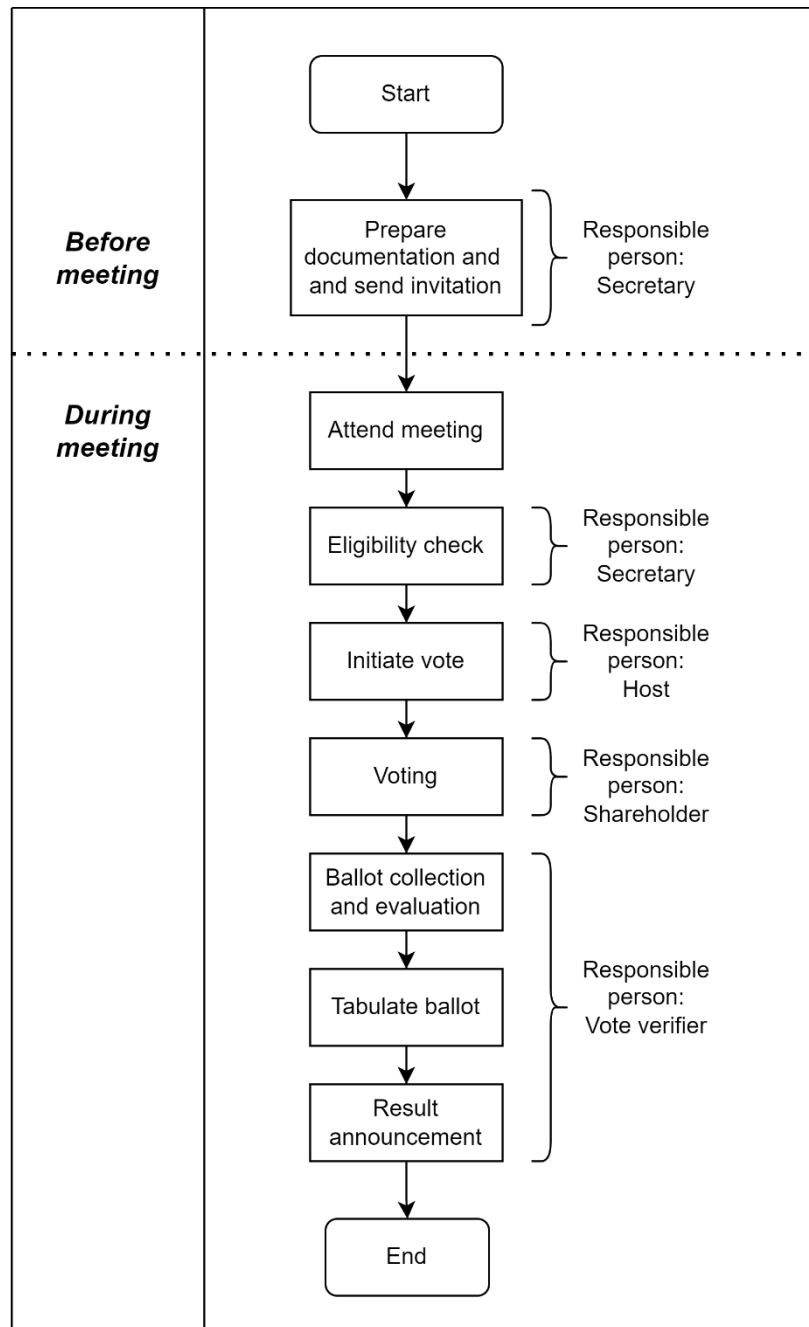


Figure 3.2: The meeting and voting processes

3.4.1 Phase 1: Requirement Elicitation Layer

Requirement elicitation collects the needs of the domain experts and stakeholders to understand and analyses the problem at hand. At AOM, HOMER is adopted on requirement elicitation (Wilmann & Sterling, 2005). The HOMER is based on the organisation metaphor of hiring a staff.

Table 3.1 shows the elicitation questions based on the HOMER questionnaire technique. For the voting system, there is a need to hire staff like a host, a secretary, a vote verifier, and shareholders. Each staff has their responsibility. First, the host initiates and hosts the meeting; the secretary helps with the documentation and invite shareholders to join the meeting; the vote verifier helps to verify each vote and count the vote; and the shareholders provide their idea and cast their vote for decision making. In a voting process, rules are essential to ensure fairness and transparency. Some of the tasks will rely on other actors to complete them tasks. For example, the host will depend on the secretary to send invitations to the shareholders, and the vote verifier will have to verify each vote in ballots.

Table 3.1: HOMER for Voting System

Question	Answer
1 If you were to solve the voting scenario, what position do you need to fill?	Host, Secretary, Vote Verifier, and Shareholder.
2 Based on the role defined in Q1:	
a. What is the purpose of this role?	- Host – To initiate and host a meeting.
	- Secretary – To work on the documentation and invitation task. - Vote Verifier – To count and verify the vote. - Shareholder – Join the meeting and cast a vote.
b. What tasks will commonly be required?	- Initiate a meeting, initiate a vote, and reveal the result.

Table 3.1 continued

c. For each task stated,	
(i) What subtasks make up this task?	- Initiate a meeting. - Participants join the meeting. - Initiate Vote. - Set voting rules. - Prepare the voting materials. - Reveal result. - Cast a vote.
(ii) What are the conditions for the agent to undertake this task?	- Participate in the meeting. - The participant must be eligible to join.
(iii) How the role achieves the task?	- Participants join the meeting. - Secretary sends the invitation to the eligible shareholder. - Reveal vote. - Verify vote ticket. - Count valid vote ticket.
(iv) Does the role depend on another role to achieve its goal?	- The host depends on the secretary to send the invitation.
	- Vote verifier depends on the voting ticket from the shareholder. - The secretary depends on the vote verifier to announce the vote result.
(v) What happens after the task is completed?	- Shareholders can achieve consensus after the vote.

Table 3.1 continued

3. What codes of behaviour are required by roles?	
(a) What codes of behaviour must be observed by all of the roles?	All meeting attendees must be shareholders or staff from the company.
(b) Is there any specific code of behaviour for a particular role?	All shareholders can only vote once. Host, secretary, and vote verifier are not eligible to cast a vote.
(c) Are there any other rules that must adhere to this study?	No

3.4.2 Phase 2: Computation Independent Modelling

In this phase, the modeller transforms the elicited answers into a goal model, role model, organisation model, and domain model. It has been reported that the models are designed for a non-technical person, which is easy to understand and comprehensive (Wilmann & Sterling, 2005). In addition, the models capture what is expected of the system by describing the context in which it operates (Bousetta, El Beggar, & Gadi, 2013). Also, the models help the analysts to examine the system and identify the problem. Furthermore, the models transcend the gap between the experts for design and development (Kardoš & Drozdová, 2010).

The details of the goal modelling, role modelling, organisation modelling, and domain modelling in the phase are described as follows.

Goal modelling

The goal model is used to illustrate the objective and motivation of the project. The goal model involves a hierarchy structure's main objective and sub-objective. Besides, the goal model models the quality goal of the system. It aims to maintain the project's vision and mission throughout the development and achieve consensus on the knowledge, problem, and solution between technical and non-technical clients. The goal model comprises five notations, as shown in Table 3.2. From the elicited answers, the modeller can map the answers into the goal models, as shown in Table 3.3 (Cheah, 2010).

Table 3.2: Notations for Goal Model

Description	Notation
Goal/ subgoal	
Quality goal	
Agent	
Relationship with a quality goal	
Relationship with actor	

Table 3.3: HOMER Mapping to Goal Model

No	HOMER's question	Goal Model
1	What happens after the task is completed?	Main goal

Table 3.3 continued

2	If you are hiring more than one staff, what position do you need to fill?	Role
3	What is the purpose of this role?	Subgoal and quality goal
4	What tasks will commonly be required?	Subgoal and goal relation
5	What are the conditions for the agent to undertake this task?	Subgoal and goal relation

Figure 3.3 shows the overview of the Voting system's goal model. The main goal of the voting is to achieve consensus among the shareholders. The host, secretary, shareholder, and vote verifier are involved in the whole process. The main goal is to reach a consensus. To reach a consensus, the host must initiate a meeting then the secretary will verify the eligible participant and send the invitation. Then, the host must begin the voting process, which requires the rules, the questions, and the voting materials. Lastly, the shareholders cast a vote, and the vote verifier handles the ticket. Once the vote is verified and counted, the secretary will announce the vote results.

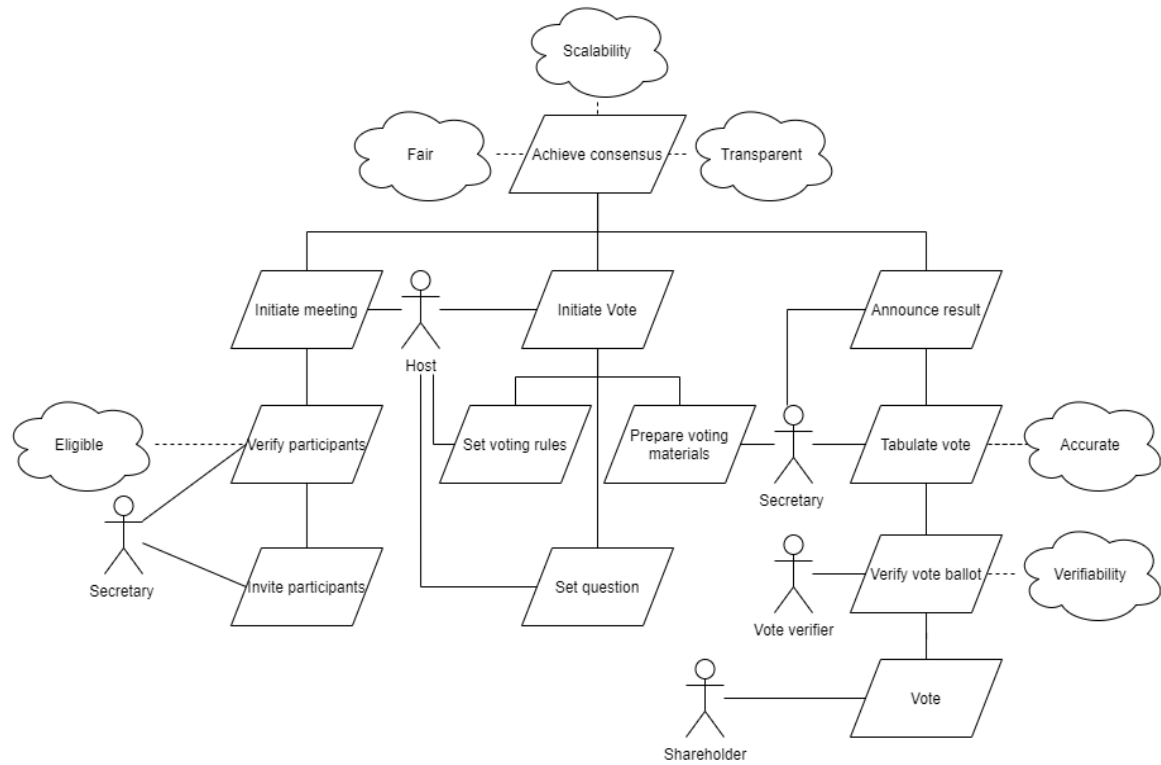


Figure 3.3: An Overview Goal Model for Voting System

Role modelling

A role model models the responsibilities of a staff member in solving the problem at hand (Sterling & Taveter, 2009). Based on the mapping guideline from Cheah (2010), the modeller can map the elicited answers into the role model.

Table 3.4 shows the overview of the role model, and Table 3.5 shows the mapping from HOMER to the role model. Table 3.6 to Table 3.9 shows the host, secretary, vote verifier, and shareholder role model. The host is the chairperson in the meeting, while the secretary assists the host and performs administrative tasks. Vote verifiers are responsible for verifying each vote from the shareholders. Meanwhile, the shareholders will make their decision by casting a vote.

Table 3.4: Overview of Role Model

Role Name	<i>Name of the role</i>
Description	<i>A brief description of the role</i>
Responsibilities	<i>Functionality or behaviour of the role</i>
Constraint	<i>The condition that will be encounter when accomplishing a goal</i>

Table 3.5: Mapping from HOMER to Role Model

No	HOMER's question	Role Model
1	If you were to solve the voting scenario, what position do you need to fill?	Role Name
2	What is the purpose of this role?	Description
3	What tasks will commonly be required?	Responsibilities
4	What codes of behaviour must be observed by all of the roles?	Constraint
5	Is there any specific code of behaviour for a particular role?	Constraint

Table 3.6: Role Model for Host

Name	Host
Description	The person who in charge of the general meeting.
Responsibility	• Initiate meeting. • Initiate vote (set questions and options). • Set voting rules. • Manage and monitor the lifecycle of the polling.
Constraint	• Not qualified to cast a vote.

Table 3.7: Role Model for Secretary

Name	Secretary
Description	A person who is responsible for assisting the host in carrying out administrative tasks.
Responsibility	• Send an invitation to the qualified shareholder to join the general meeting. • Collaborate with the host and ensure the meeting are well-organized. • Keeping up with paperwork (included vote result) and management. • Records the voting rules that are agreed upon among the host and shareholders. • Prepare vote material such as a form or paper ballot. • Tabulate vote in the public to ensure transparency.
Constraint	• Not qualified to cast a vote.

Table 3.8: Role Model for Vote Verifier

Name	Vote verifier
Description	A person who is responsible for verifying the vote.
Responsibility	• Verify the vote. • Cross-check each vote and prevent duplicate votes from the same shareholders.
Constraint	• Voting is performed. • He/She is not qualified to cast a vote.

Table 3.9: Role Model for Shareholder

Name	Shareholder
Description	A shareholder who owns at least one share of the corporation.
Responsibility	• Attend a general meeting. • Brainstorm and discuss strategies to achieve the company's goal and objective. • Decide by casting a vote. • Obey the voting rules stated by the host.
Constraint	• Eligible to join the general meeting. • Eligible to cast a vote.

Organisational Modelling

The organisation model models the structure of a group of actors on goal accomplishment. Zambonelli (2011) discovered that relationships between humans and organisations are categorised using control, peer, benevolence, dependency, and ownership. Control will define the role's authority within a system; peer states the equal authority of the roles; benevolence describes the interest shared among the roles; dependency specifies how the roles depend on each other to achieve the goal; and ownership determines the delimit corporate limits.

As shown in Figure 3.4, the organisation of the voting system is presented. It clearly shows that all the actors involved in the voting process know each other.

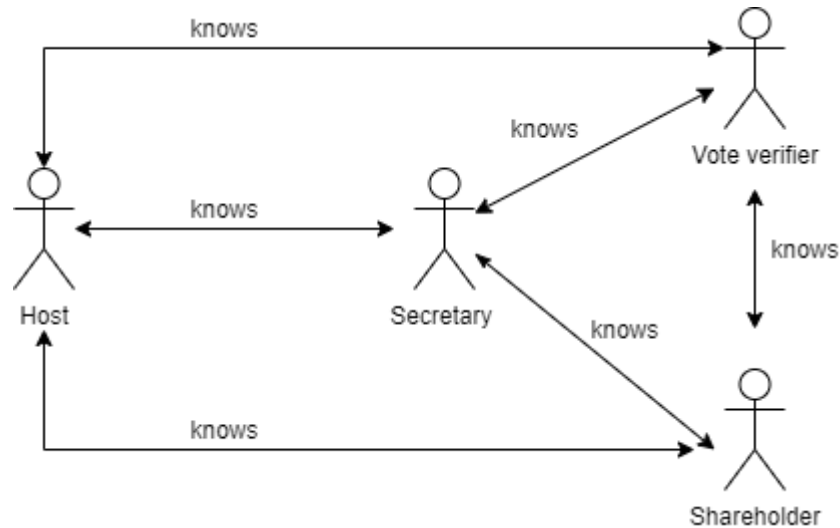


Figure 3.4: Organisational Model for Voting System

Domain modelling

The domain model treats the shared knowledge as domain entities and displays the agent's relationship and the domain entities during any circumstances.

In handling the blockchain-enabled voting system, the host initiates the meeting, and the secretary assists the meeting by sending an invitation to the shareholder, as shown in Figure 3.5.

During the meeting, which contains voting rules, the vote is started. The shareholder will vote on the paper ballot, and the vote verifier will verify each vote. Then, the result will be tabulated and announced at the end of the voting.

The conceptualization phase of the voting system is presented in the previous description. The modeller will elicit the blockchain use case through the e^3 value model upon understanding the problem.

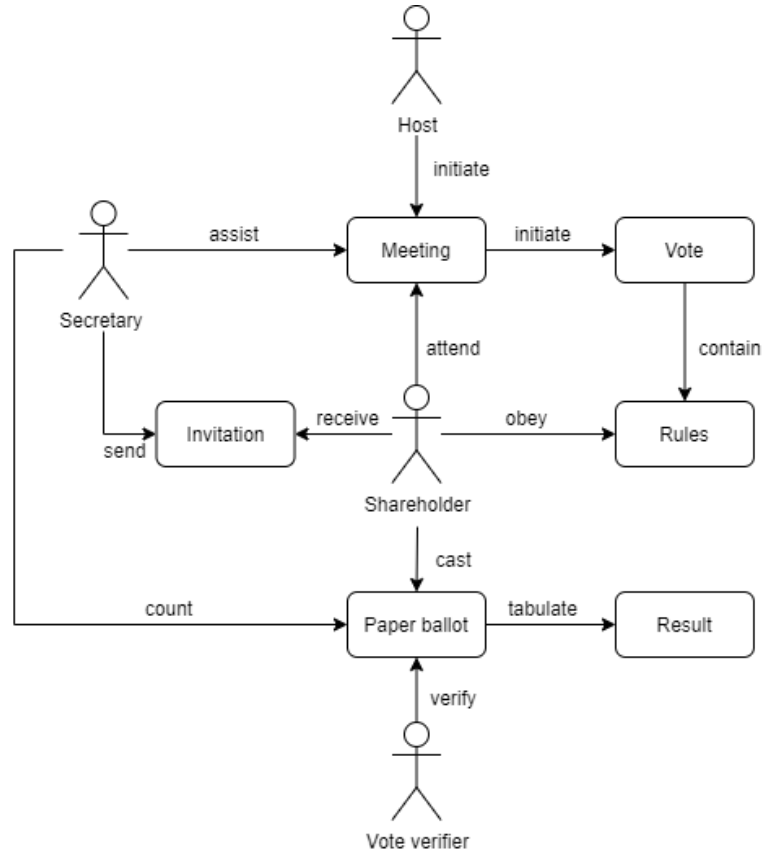


Figure 3.5: Domain Model for Voting System

3.4.3 Phase 3: Early Identification of Blockchain Use Case

Assuming that one might develop a decentralised voting system based on the problem understanding as described in goal model, role model, organisation model and domain model, a primary question would arise: How to decide the decentralised voting system? In other words, how to determine the blockchain use case?

We adopted the work from Poels, Kaya, Verdonck, and Gordijn (2019) to elicit the blockchain requirements by using e^3 value model. The e^3 value model represents the governance operation process and behaviour of the existing system and then aids in identifying the opportunity for blockchain-enabling entities such as cryptographic transactions and smart contracts. In addition, e^3 value models can fulfil the requirements of

replacing intermediaries, a market structure that works in peer and immutable transactions for a blockchain business case.

To fulfil one of the requirements, there is a need to define the middleman. One of the beauties of blockchain technology is to enhance trust by removing the middleman. The middleman is the entity that holds the authority to impose rules and regulations on the involved parties (Poels et al., 2019). Besides, due to the immutability characteristics of blockchain technology, the person who handles data could be replaced.

In this case, the secretary and vote verifier is the middleman. Secretary is the assistant of the host who helps to send invitations and documentation during the meeting. At the same time, the vote verifier is the person who handles the vote count, which requires being faithful. Thus, the secretary and vote verifier could be replaced. However, the design for middleman replacement is very subjective, and the design decision is based on the modeller.

Figure 3.6 shows the value exchange between the actors when handling the voting system based on conceptual independent modelling. Four main actors perform value exchange in the traditional voting system, and most of the processes will go through the secretary. The secretary will announce the rules set by the host and send invitations to the shareholder. The casted vote will pass to the vote verifier to perform verification and calculation. Lastly, the results will return to each actor.

Figure 3.7 shows the replacement of the intermediary with an aggregator (secretary and vote verifier) on a blockchain-enabled voting system. In this case, the aggregator will handle the rules, invitation, and vote and how the main person of the voting system interacts with the aggregator.

Based on the requirements stated by Poels et al. (2019), the e^3 value model based on Figure 3.6 and

Figure 3.7 can highlight the following points:

1. **Eliminate middleman (Requirement 1):** the aggregator acts as a middleman to handle the ballot.
2. **Establish trust (Requirement 2):** The parties need to be trusted to handle the ballot and records.
3. **Immutable data history (Requirement 3):** The aggregator will keep track of the ballot records.

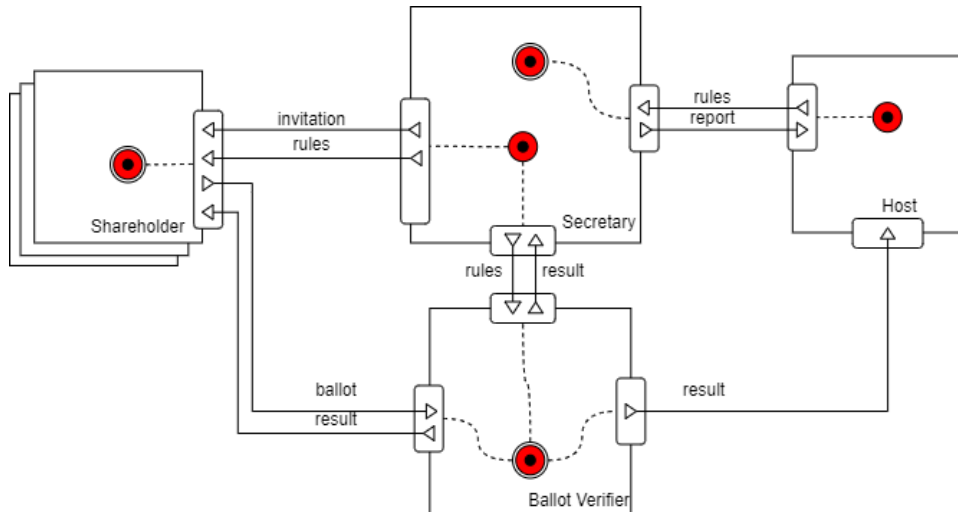


Figure 3.6: e^3 value Model for Traditional Voting System

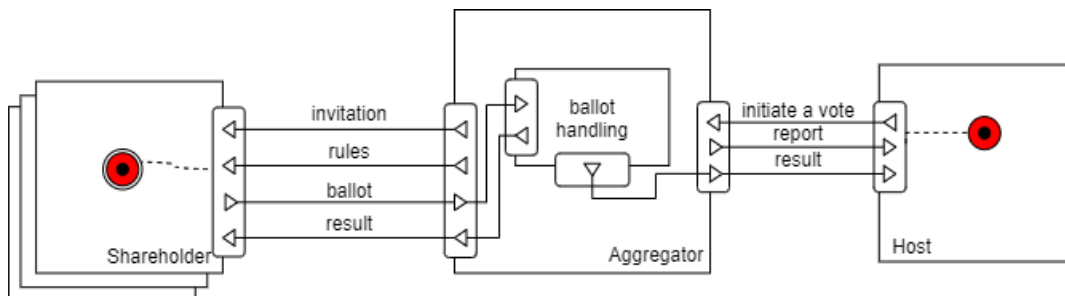


Figure 3.7: e^3 value Model for Blockchain-enabled Voting System

Once the blockchain use case is identified, the modeller can start to model the design aspect of the blockchain application.

During the design phase, the modeller must transform the actors from the conceptualization domain modelling phase and early identification of the blockchain use case into a dedicated agent. The transformation is done through an agent-role mapping diagram. The agent-role mapping diagram is modelled to map the human actor into a software/human agent.

Figure 3.8 shows the mapping of the agent role model. The mapping descriptions for each agent are listed below:

1. The administrator agent will be the host

- As an administrator agent replacing the host's position, he/she is responsible for initiating a vote by setting questions and options. Then, manage and monitor the lifecycle of the polling. However, he/she is not qualified to cast a vote.

2. Voter agent will be the shareholder

- The voter agent may decide by casting a vote and obeying the predefined rules stated in the smart contract.

3. Blockchain agent will handle the task of the secretary,

- Blockchain agent is a global ledger that will record all transactions made by the human agent (administrator agent and voter agent)

4. Smart contract agents will be part of the secretary and vote verifier to execute the rules.

- Smart contract is responsible for checking all predefined rules for every transaction.

5. Wallet agent will be a vote verifier.

- Wallet agent will hold the cryptocurrency and the externally owned account address. The externally owned account address will be recorded in the blockchain when a transaction occurs. Human agents may pay the gas fee through the wallet agent when necessary.

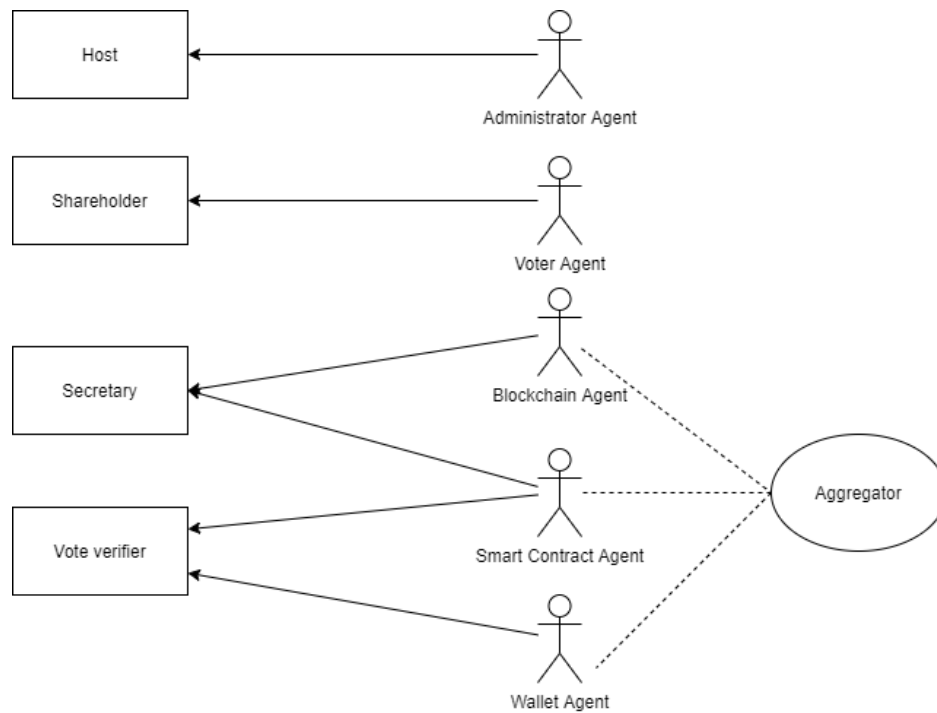


Figure 3.8: The Agent Role Model Mapping

With agents, the design phase involves modelling the knowledge model, interaction model, scenario model, and behaviour model. These models will capture the system's specification design by defining the agents' activity and environment.

3.4.4 Phase 4: Platform Independent Modelling (PIM)

In this phase, the blockchain properties are designed for the use case by using the knowledge model, interaction model, scenario model, and behaviour model.

Knowledge Model

The knowledge model shows the knowledge or information of the agent required to perform their behaviour. Knowledge models are an ontology that provides a knowledge framework for the problem domain.

Figure 3.9 shows the information shared among the agents using the knowledge model. Each human agent (voter agent and administrator agent) will hold a wallet comprising of the public key, private key, and Ethers. Each voter agent can cast a vote that the smart contract agent handles. The smart contract agent will record the voters and vote details. Every transaction will then be recorded in a block and handled by the blockchain agent.

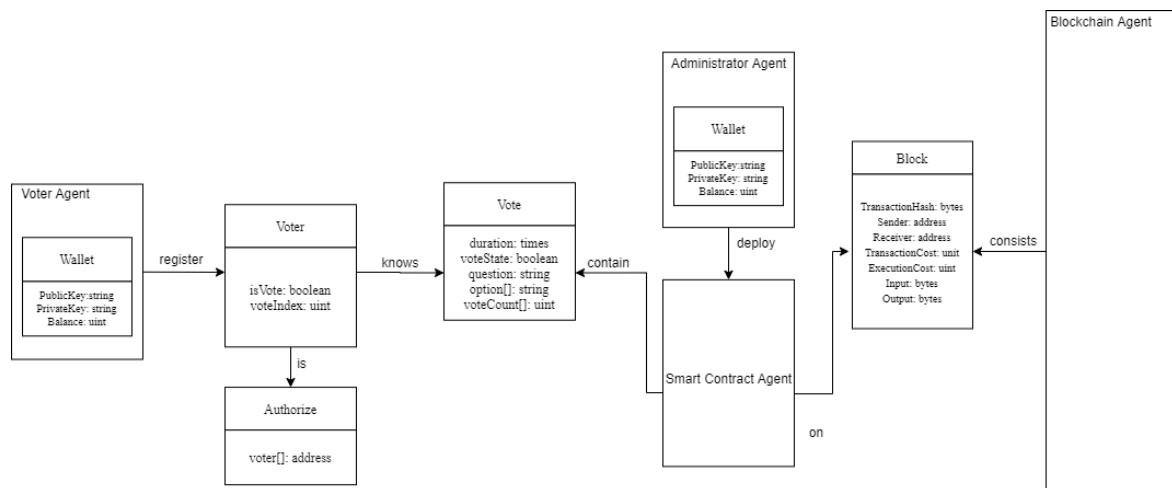


Figure 3.9: Knowledge Model for Blockchain-enabled Voting System

Interaction Model

The interaction model defines the protocol between the agents, such as the purpose, initiator, responder, inputs, outputs, and processing. It shows the potential information exchange or request in interaction patterns between agents in sequential order. A stakeholder can understand the agents' interaction and message flow from the interaction model. Figure

3.10 shows the interaction model for the blockchain-enabled voting system. The web agent will display the questions and options to the voter agent. Then, the voter agent may decide on the web agent. Once the option is selected, the web agent will invoke the wallet agent to request a signature as authentication. Next, the smart contract will be auto-generated to store the decision on the blockchain by creating a transaction. The blockchain agent will return the unique hash to the wallet agent. After that, the web agent will display the message “decision recorded”.

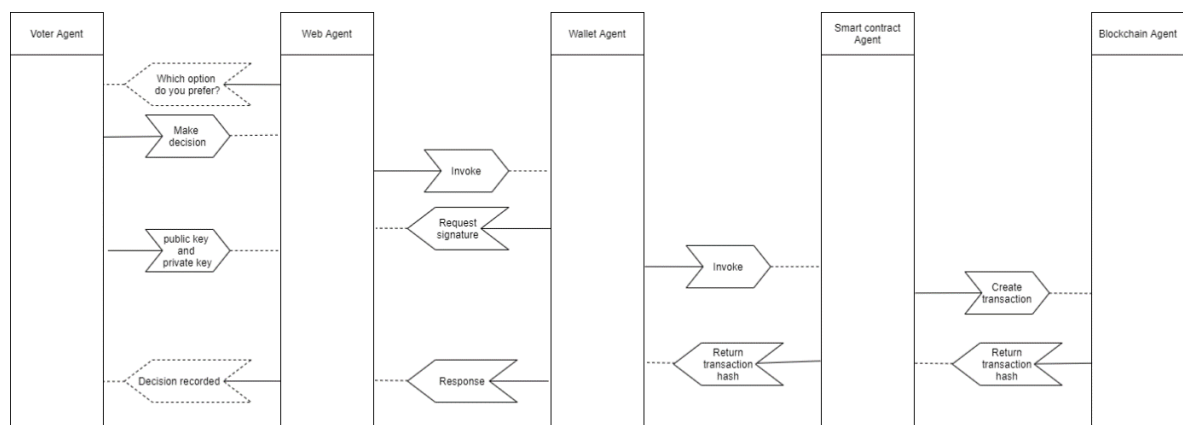


Figure 3.10: Interaction Model for Blockchain-enabled Voting System

Scenario Model

This model contains the steps and strategy of each agent to accomplish a goal in sequential order. The scenario model follows a schema that will display the goal, the initiator, the event to be a trigger, and a list of steps. Each step will be written based on the agent's reaction to the environment and the environment's response to the agent. Table 3.10 shows the scenario while removing central authority by the voter agent.

Table 3.10: Blockchain-enabled Voting System Scenario Model

Scenario 1	
Quality Goal	Remove central authority

Table 3.10 continued

Initiator		Voter Agent		
Trigger		Store vote detail on blockchain		
Description				
Condition	Step	Activity	Agent types/role	Immutable/mutable resources
	1	Launch the web application		Web application - mutable
	2	Receive invitation		
	3	Select decision		
	4	Pay gas	Wallet agent	
Vote state is true, and the voter must be eligible	5	Manage vote count	Smart contract agent	
The condition passed	6	Return transaction hash	Smart contract agent	
	7	Redirect to vote screen		

Behaviour Model

From the scenario model, we can proceed to the behaviour model. The Behaviour Model explains the behaviour of an agent in an execution cycle (Sterling & Taveter, 2009). An agent's behaviour is defined based on perceptions and knowledge in terms of rules. A start event will trigger a rule when the condition is true, updates the agent's mental state, and

sends a message or performs an action. The agent may perceive the message another agent has sent or a physical action taken by another agent.

Figure 3.11 shows the Behaviour model for the blockchain-enabled voting system.

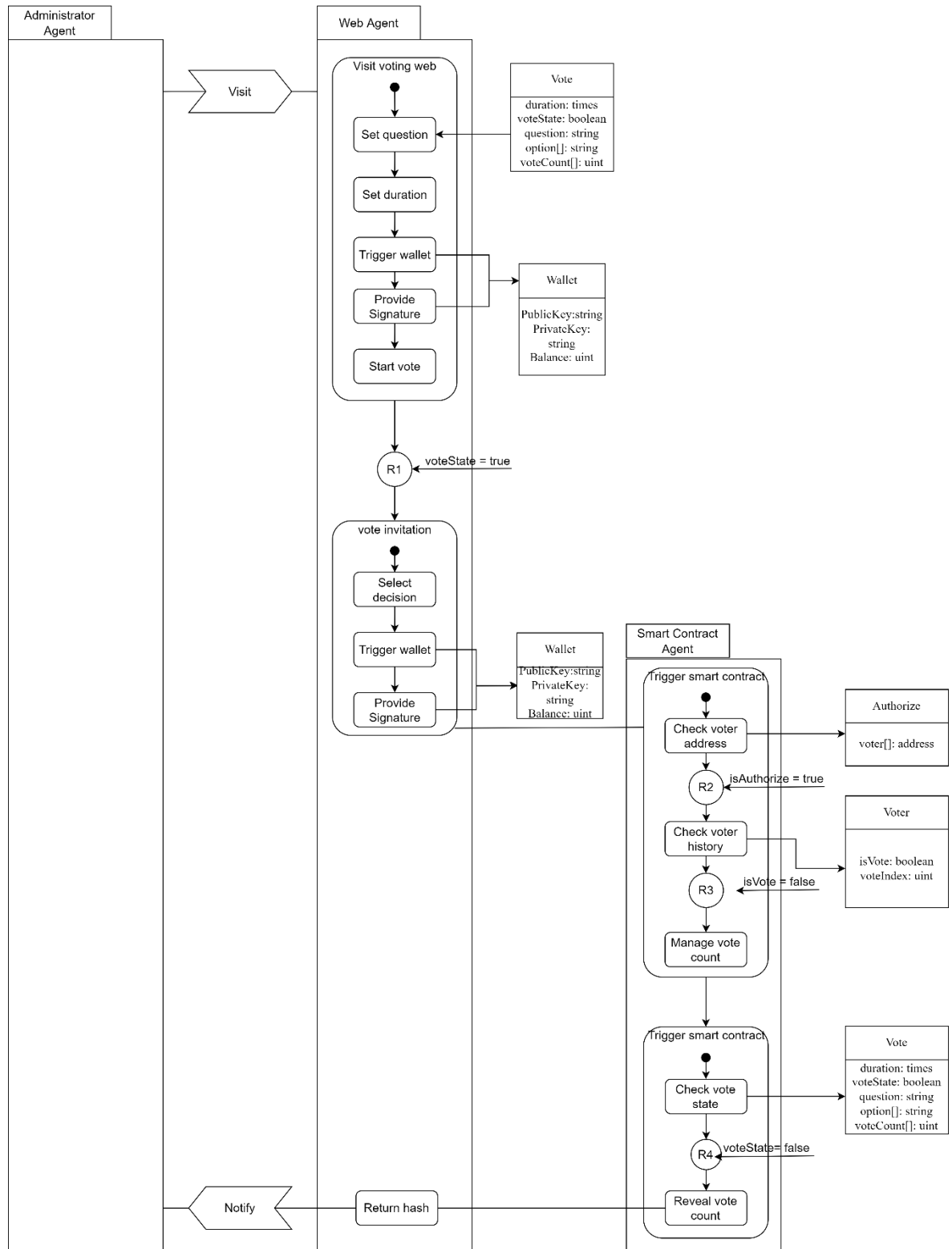


Figure 3.11: Behaviour Model for Blockchain-enabled Voting System

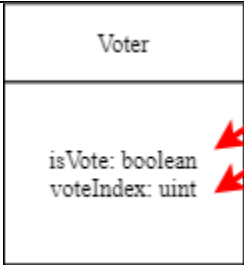
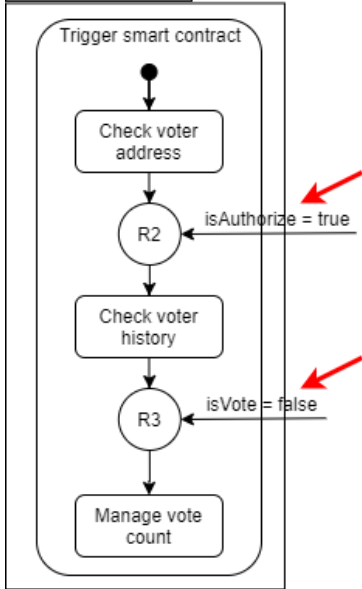
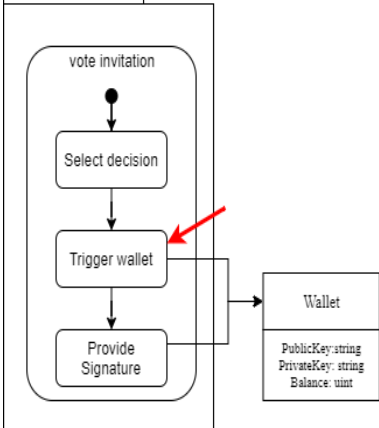
The steps from Phase 1 to Phase 4 aid in plan and designing a blockchain application. In phase 1, the modeller understands and analyses the problem through the HOMER technique. Then, the modeller maps the answer from HOMER to the computation-independent modelling. To identify and design a blockchain use case, the modeller constructs the e^3 value model to extract the blockchain requirement. Once the blockchain use case is confirmed, the modeller will design the blockchain component through platform-independent design and modelling. From the design model, the modeller can continue implementing the blockchain-enabled voting system by using the transformation rules as in the platform-specific model.

3.4.5 Phase 5: Platform Specific Model

The transformation rules are presented to map the design models to code. Ethereum platform is selected to adopt the transformation rules. Ethereum smart contract is written in Solidity, which is an object-oriented language.

The transformation rules are derived from (Cheah, 2010). The transformation rules guide the programmer to code based on the model. The pointed arrows indicate the corresponding code. The first and second guidelines transform the model into Solidity code. The programmer could identify the smart contract attribute or property from the first rules based on the knowledge model. The second guideline shows that the smart contract's logic or condition is required based on the behaviour model. The third guideline transforms the model to JavaScript, which allows the web agent to communicate with smart contracts. This rule aids in identifying the rules or logic for the web application.

Table 3.11: The Transformation Guideline

Model Artifacts	Corresponding Code	Samples
	Contains smart contract attributes or property. Transform the model to solidity code.	<pre>boolean isVote; uint voteIndex;</pre>
	Contains the smart contract conditions.	<pre>function vote() { require(msg.sender == shareholder); require(isVote == false) //then update vote count }</pre>
	Contains the action in the web agent that handle in javascript.	<pre>loadAccount: async () => { App.account = web3.eth.accounts[0] },</pre>

3.5 Chapter Summary

In this chapter, extended AOM provided a structured and guided process to analyse and design a blockchain-enabled voting system. The extended AOM is elaborated through a walkthrough example of a blockchain-enabled voting system. It started by analysing the problem from the traditional voting system at the requirement elicitation phase. Then the modeller map all the requirement to computation independent modelling. Next, the modeller identifies the blockchain use case using the e^3 value model to present the three requirements of blockchain technology. After the blockchain use case is identified, the design of the blockchain application will take place at the platform-independent modelling. Transformation guideline is derived based on the model artefacts.

The next chapter evaluates the potential usage of extended AOM through a walkthrough example of a blockchain-based lottery application.

CHAPTER 4

PROTOTYPING AND PILOT TESTING OF THE EXTENDED AOM

4.1 Overview

This chapter presents a case study to demonstrate the feasibility of the extended AOM. Section 4.2 presents the elaborated case study, a blockchain-based lottery application. Section 4.3 presents the modelling of blockchain-based lottery applications through extended AOM. It covers the complete extended AOM, the requirement elicitation, computation independent modelling, early identification of blockchain use case, platform independent-modelling, and platform-specific model. Section 4.4 presents the lesson learnt. The chapter is concluded in Section 4.5.

4.2 Elaborate Case Study

A lottery is a type of game that provides players with an opportunity to win big prizes. The lottery differs from gambling which does not require any technique to win the game but depends on luck. Every player wishes to become an instant millionaire by using a small bet. Players buy a lottery ticket from a lottery counter. The winner will announce through social media, such as newspapers, radio, or through official lottery applications. Although lottery industries have existed for decades, they suffer from trust issues (Liao & Wang, 2017). The lottery industry faced trust issues among the player, such as the ticket, prizes, and winner. Hence, many blockchain-enabling lottery applications have been developed. Some of these applications are FairLotto (Liao & Wang, 2017) and DeLottery (Jia, Chen, & Li, 2019). Due to the game's popularity, we present how the blockchain-based lottery application can be developed through the extended AOM.

4.3 Modelling of Blockchain-based Lottery Application through Extended AOM

As mentioned in Chapter 3, there are 5 phases in modelling blockchain-enabling applications through extended AOM.

4.3.1 Phase 1: Requirement Elicitation of Blockchain-based Lottery Application

Based on the current works and lottery industry (Liao & Wang, 2017; Saichua, Khunthi, & Chomsiri, 2019), the elicitation answers are derived in Table 4.1.

Table 4.1: Elicitation Question and Answer

Question	Answer
1 If you were to solve the lottery application, what position do you need to fill?	1. Government 2. Lottery agency 3. Middleman 4. Player 5. Banker
2 Based on the role defined in Q1:	
a. What is the purpose of this role?	1. Government - Regulate the lottery industries. 2. Lottery agency: - A department that will manage the entire lifecycle of the lottery game and rules the company. 3. Middleman - The middleman deals with the sales of lottery tickets and communicates with players to encourage them to buy tickets. 4. Player

Table 4.1 continued

	- Buy a lottery ticket and win prizes. 5. Banker Monitor and manage the financial flow of the lottery agency.
b. What tasks will commonly be required?	1. Government - Set rules and regulations of the lottery industry. - Monitor the lottery industry. 2. Lottery agency: - Set, announce, and follow the game's rules and regulations. - Pick a winner from the player. - Verify and validate the lottery ticket of the winner, then distribute the prize. 3. Lottery intermediary - Organize and manage the selling of one or more lottery tickets to players. - Provide games details on different lottery games and rules to the players. - Support and broaden lottery distribution, devise sales and marketing strategies. - Process the ticket sold and report the amount to the lottery agency. 4. Player - Read and understand the rules and agreements. - Buy/rebuy a lottery ticket. - Wait for the lottery result.

Table 4.1 continued

	- (Winner) Redeem the prize at the lottery company counter by using the ticket. 5. Banker - Manage lottery agency bank account (opening and closing account) Handle monthly transactions by the lottery company.
c. For each task stated,	
(i) What subtasks make up this task?	1. Lottery agency: - Marketing and promote the lottery game. - Establish an image and trust of the company. - Set, announce, and follow the game's rules and regulations. - Pick a winner from the list. - Verify and validate the lottery ticket before distributing the prize. 2. Player: - Buy a ticket from the official lottery company. - Redeem the prize at the lottery company by showing the ticket. 3. Banker - Depends on the transaction activity of the lottery agency.
(ii) What are the conditions for the agent to undertake this task?	1. Lottery agency and intermediary must follow the law stated by the government. 2. Lottery agency and intermediary must follow the rules and regulations stated to get trust from the player.

Table 4.1 continued

	3. Player must keep the original ticket for the prize redemption. 4. Banker must work on the transaction history precisely to generate an accurate report for the lottery company.
(iii) How the role achieves the task?	1. Lottery agency must protect the reputation of the company from developing trust with the players. 2. The intermediary must sell the ticket from the official platform. 3. Player must buy a ticket from the official counter of the company, keep the lottery ticket, and redeem the prize at the official counter.
(iv) Does the role depend on another role to achieve its goal?	1. Lottery agency depends on an intermediary to sell the ticket. 2. Lottery agency depends on bankers to get an accurate report. 3. Intermediary depends on the player to achieve sales target. 4. Player depends on an intermediary to purchase a ticket. 5. Winner depends on the lottery agency to get the prizes.
(v) What happens after the task is completed?	The winner of the game will get the required prize mentioned in the rules and regulations. The business of the company will grow by complying with the rules and regulations.
3. What codes of behaviour are required by roles?	

Table 4.1 continued

(a) What codes of behaviour must be observed by all of the roles?	Must be trustworthy, transparent, reliable, and integrity.
(b) Is there any specific code of behaviour for a particular role?	The company must handle scalability. The banker must be mindful to handle lottery company transactions.
(c) Are there any other rules that must adhere to this study?	None

4.3.2 Phase 2: Computation Independent Modelling

Goal Modelling

Figure 4.1 illustrates the goal model for operating a lottery industry. The roles involved in this lottery agency are government, banker, a lottery agency, lottery intermediaries, and player. To successfully operate the lottery industry, the full process consists of reliability, integrity, ethics, and scalability. The government will need to monitor by providing and stating the necessary law. The law states must be fair, reasonable, and transparent. Next, the banker is responsible for monitoring and managing the finances of the lottery company. The financial flow must be given accurately. The lottery agency must manage the company and guide the agents to achieve sales targets. Lastly, the player is the "client" of the industry that must have faith to support the lottery industry to win a fortune.

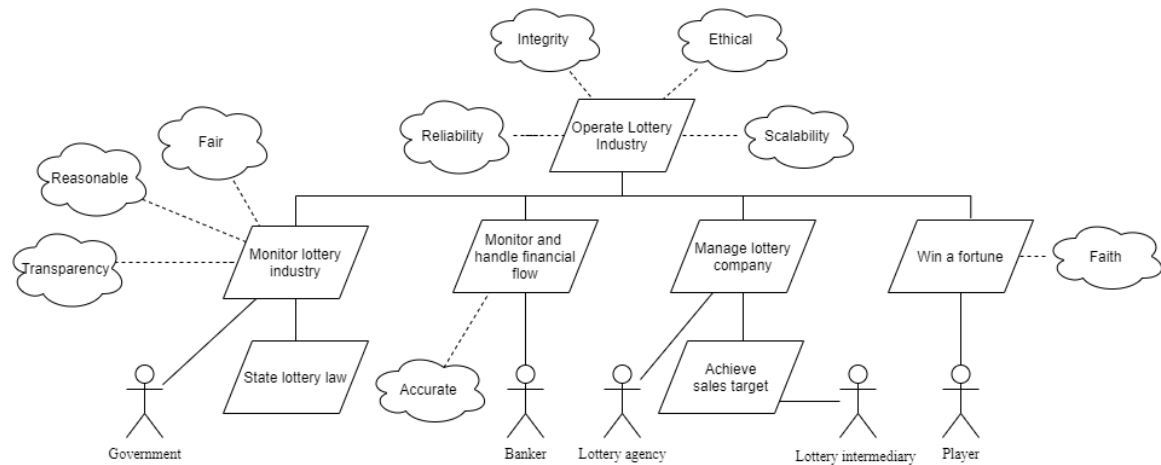


Figure 4.1: The Goal Model of Govern Lottery Industry

Figure 4.2 shows the subgoals of the lottery agency in managing the lottery company. The company must be reliable, integrity, ethical, transparent, and handle scalability. The company should be reliable and have the integrity to gain the players' trust. The process of the games should be ethical and must handle the scalability issue when the number of players increases. Rules and regulations of a lottery company should be transparent and specific to protect the company and ensure the business practices are systematic. The lottery intermediaries play an essential role in selling tickets to the players and achieving sales targets. Next, the actors must report daily sales accurately to the company. At the end of the game, the lottery agency will select a winner from the player. The agency must verify the lottery ticket from the winner and distribute the prize.

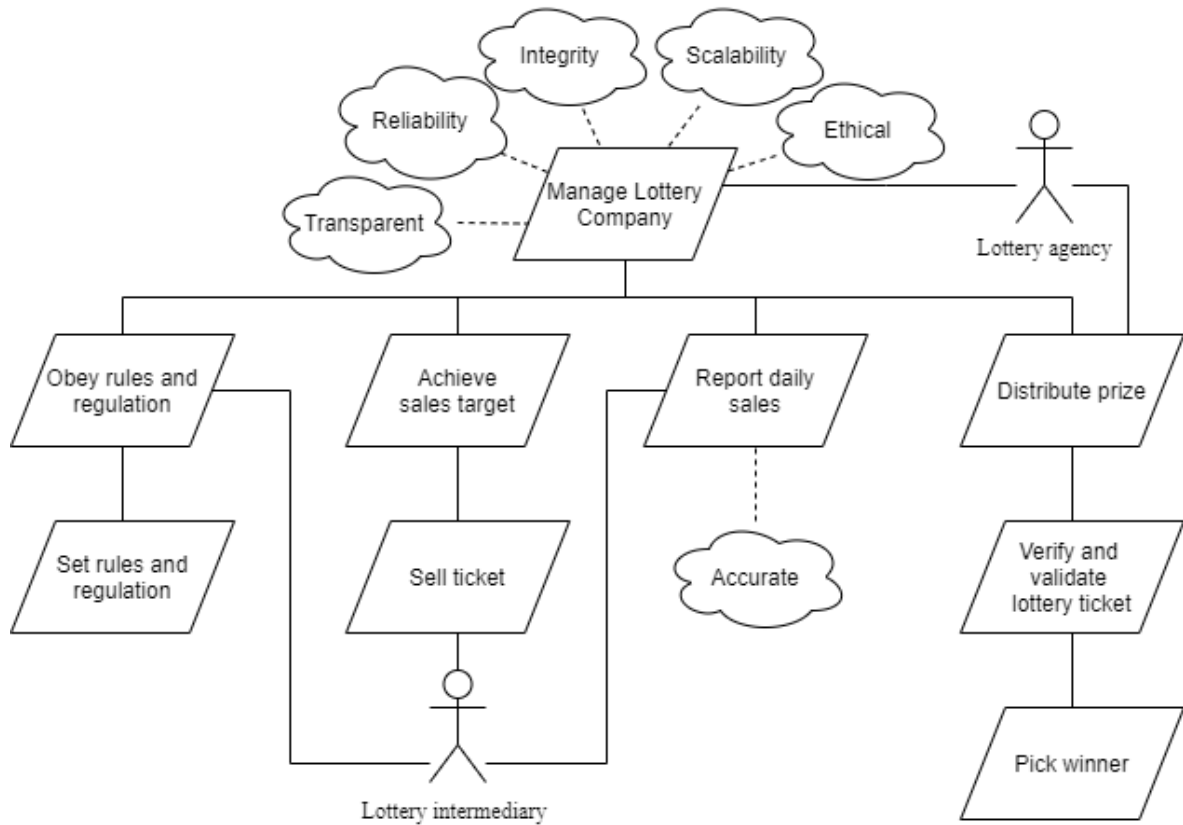


Figure 4.2: Goal Model for Manager Manage the Lottery Company

Figure 4.3 illustrates the player who aims to win a fortune by purchasing a lottery ticket. Before purchasing the lottery ticket, the player must understand and obey the rules and regulations stated by the lottery company. When the company officially announces the winner, the winner may redeem the prize at the company's counter by showing the lottery ticket to the manager. In purchasing a lottery and winning a fortune, the player must have faith in the lottery company and integrity by purchasing the ticket at the official lottery counter.

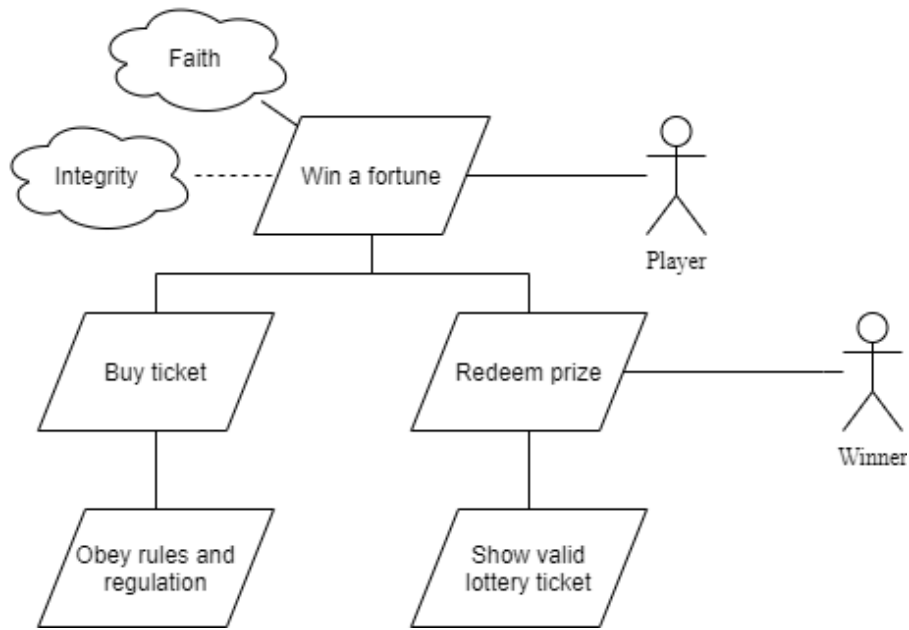


Figure 4.3: Goal Model for Winning a Fortune by the Player

Role Modelling

Table 4.2 shows the role model for the government; Table 4.3 shows the role model for the banker; Table 4.4 shows the role model for the lottery company manager; Table 4.5 shows the role model for the lottery agency; and Table 4.6 shows the role model for the lottery game player.

The government is a group of people that will govern the lottery agencies and monitor their activity. The government should construct the most appropriate law for lottery agencies. Banker is a group of people to offer financial and banking services. They handle all the transaction activities and generate monthly reports for the lottery industries. A lottery agency is a department responsible for managing and operating the company. They must follow the rules and laws stated by the government and the company. Lottery intermediaries will help the company to sell the ticket to the players through an official counter or platform.

Table 4.2: Role Description of Government

Name	Government
Description	A group of people that govern the lottery agencies.
Responsibility	• Define the lottery law. • Monitor the lottery agencies' activity. • Collect tax
Constraint	• Construct reasonable and the appropriate law for lottery agencies.

Table 4.3: Role Description of Banker

Name	Banker
Description	A team of people that provide financial and banking services to lottery agencies.
Responsibility	• Handle transaction activity. • Manage lottery agency bank account (opening and closing account) • Provide suitable financial suggestions. • Generate financial report to the lottery agency.
Constraint	• The transaction activity must be accurate. • The specific bank must undergo the transaction (credit/debit).

Table 4.4: Role Description of the Lottery Company Manager

Name	Lottery agency
Description	A department that manages the operation of the company.
Responsibility	• Set, announce, and obey rules and regulations. • Select a winner. • Verify and validate the winning lottery ticket. • Distribute the prize to the winner.
Constraint	• Select winner based on the rules and regulations stated.

Table 4.5: Role Description of the Lottery Intermediary

Name	Lottery intermediary
Description	The person who serves the lottery company and act as the middleman of manager and player.
Responsibility	• Help the company sell a lottery ticket. • Obey the rules and regulations set by the company. • Calculate and report the number of tickets sold and total bets in a game. • Boost company sales.
Constraint	• The ticket must sell at the official lottery counter or platform. • The report of tickets sold should be valid and immutable.

Table 4.6: Role Description of the Lottery Game Player

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

Organisation Modelling

Figure 4.4 shows the organisation models for the lottery industry that illustrates the relationship and structure of the government, lottery intermediaries, bankers, lottery intermediaries, player, and winner. The government controls the lottery agency by using the law. The agency will cooperate with the banker to manage the company's financial flows. Next, the lottery intermediaries are controlled by the lottery agency to obey the rules and regulations of the company. The lottery intermediaries know the player when the player

purchases tickets from the agent. Lastly, the winner is inherited from the players the lottery agency selects.

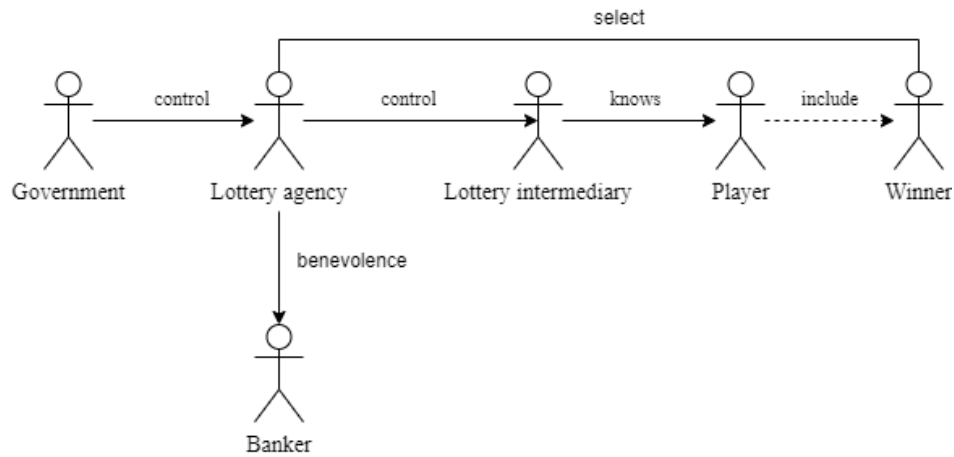


Figure 4.4: Organisation Model for Lottery Industry

Domain Modelling

Figure 4.5 shows the domain model that illustrates the relationship between the actor and entities when an event occurs. In the entire cycle of lottery operation, the lottery agency must follow the law stated by the government. The government will monitor the lottery agency to prevent the agency breaks the law. Besides, the lottery agency, intermediary, players, and the winner must enforce the company's rules and regulations. The transaction will be done between the intermediary and the player by ticket trading. The lottery intermediaries will calculate the total sales and generate a report for the agency.

The banker handles the transaction flows of the company and generates a financial report for the agency. Then, the player will wait for the result announcement based on the time frame stated in the rules and regulations. Once the winner is announced, the winner can collect the prize at the official counter. However, the player must show the lottery ticket to the lottery agency as proof. The agency is responsible for verifying and validating the ticket before handing the prize to the winner.

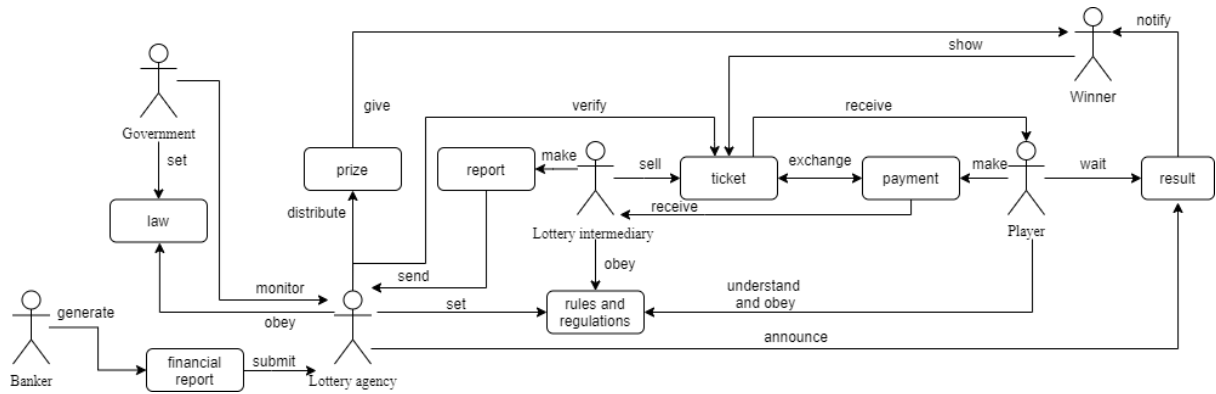


Figure 4.5: Domain Model for the Lottery Case Study

4.3.3 Phase 3: Early Identification of Blockchain Use Case

In this phase, the e^3 value model can highlight the needs of blockchain use cases through value exchange between actors. From the analysis, the e^3 value model allows the modeller to eliminate middlemen and identify the immutability of data transactions and the data shared among the market structure. Two e^3 value models will be conducted to be compared the required actors. The first e^3 value model will represent the traditional lottery industry, and then some of the actors will be replaced by an “aggregator” as shown in the second e^3 value model. This aggregator is an agent that will replace the middleman. Then, all the data flows in the e^3 value model represent the traceable and transparent data that the ticket bought by the players. The market structure is represented by using market segments.

Figure 4.6 shows the e^3 value model of the traditional lottery industry. The model shows that it involves more than one party to exchange information to achieve the goal. Based on the conceptualization domain modelling, the problem is noticeable. The lottery company develops and preserves its reputation by investing in marketing and advertising. The company needs to handle transparency, scalability, reliability, and integrity. Doing this aids in gaining the players’ trust and selling more tickets. Then, the company must handle scalability when the number of players increases. The company will need a complete and

strict standard operating procedure to follow up on daily sales and tickets. The lottery agency must sell the ticket through an official platform.

Furthermore, the lottery agency must be accurate and reliable in reporting daily sales to the agency. The agency and player will always concern about the validity of the ticket. Thus, the agency will verify and validate the winning ticket before distributing the prize.

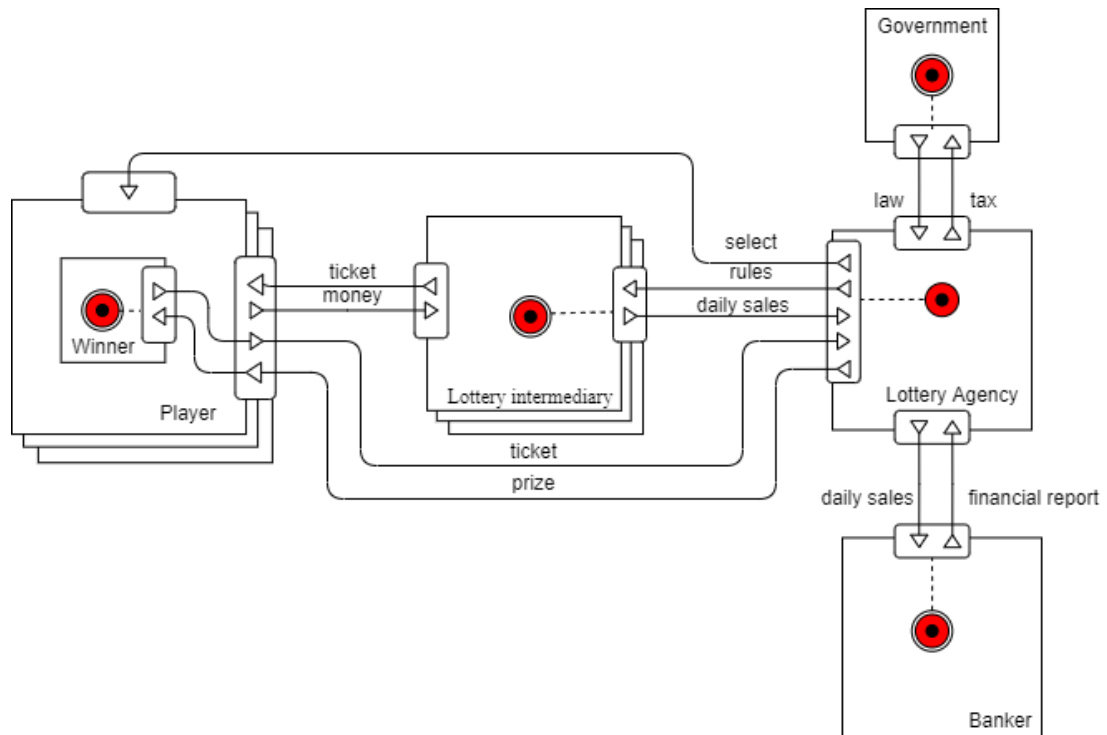


Figure 4.6: e^3 Value Model for Traditional Lottery Company

Liao and Wang (2017) emphasize that the current lottery games incite the untrust of the players due to the black box operation and services. Doubt arises regarding the transparency and fairness of the lottery game. The questions raised by the players are: Is the lottery ticket valid? Is the randomness and security of the drawing guaranteed? Is there a way to find out who won after the drawing? Is the prize allocation equal and accurate? Why are so many lottery-related corruptions, frauds, and controversies published regularly? Thus, these trust and transparency issues must be solved in a more novel way.

Thus, a lottery case study will continue to model using e^3 value models to conduct early identification of blockchain. According to Poels, et al. (2019), a blockchain business case requires removing the middleman, and providing trust and immutable data.

In this case, the aggregator will replace the lottery intermediary, government, and bankers. Government and bankers are less directly involved in the activity. Considering blockchain characteristics, it manages transaction and data records that could take over banker's tasks. Besides, the smart contract can enforce the rules and task as the government and the lottery intermediary does based on the predefined rules.

Figure 4.7 shows the value exchanged between the lottery agency, player, winner, and aggregator. The aggregator is an actor that will keep track of the tickets sold and how many transactions are made. Additionally, the aggregator is responsible for monitoring the rules and checking conditions throughout the process. The lottery agency will provide the ticket to the aggregator; then, the aggregator will sell the ticket to the player. The player will pay the ticket money to the aggregator and get the lottery ticket, and then the aggregator will manage all the tickets sold and the ticket money. Once the winner is selected, the winner will show the ticket to the aggregator to request validation. The aggregator will verify the ticket, distribute the prize to the winner, and commission it to the lottery agency.

Based on the requirements stated by Poels et al. (2019), the e^3 value model can highlight the following points:

1. **Eliminate middleman (Requirement 1):** The aggregator acts as a middleman to handle the ticketing and transaction.
2. **Establish trust (Requirement 2):** The parties need to be trusted to handle the ticket and transaction.

3. **Immutable data history (Requirement 3):** The aggregator will keep track of the ticket and transaction.

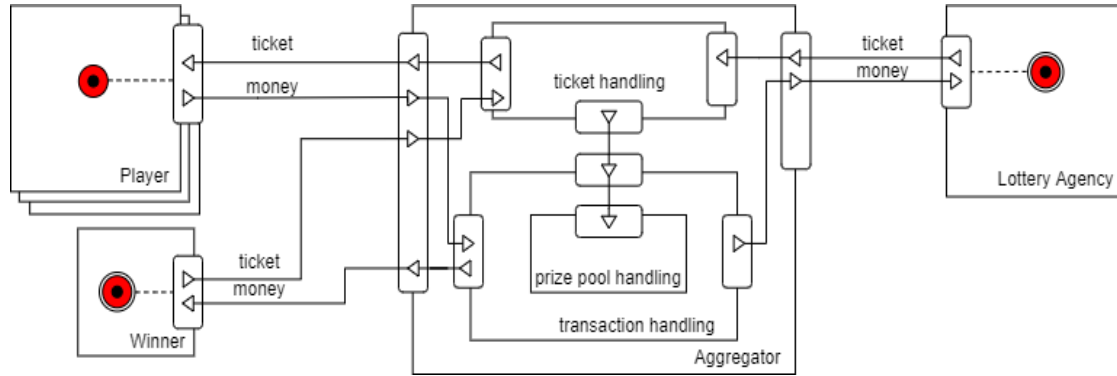


Figure 4.7: e^3 Value Model for Lottery Company

Agent and role mapping diagram

First, the agent and role mapping diagram are derived. The agents are mapped to the roles of the lottery system, as shown in Figure 4.8. The manager agent was designed to replace the lottery agency, the player agent to replace the player, and the winner agent to replace the winner. In this case study, the smart contract agent is the middleman between the manager agent and the player agent, and it could replace the government, banker, and some of the tasks of the lottery agency. It could also help to enforce the rules and regulations systematically and unmistakably to establish trust.

Additionally, the smart contracts agent holds the player agents' tokens from the game start. When achieving the condition and the game ends, the smart contract agent automatically transfers tokens to the winner agent and manager agent. By doing this, the application will concretely the trust of the player agent to bet on the game.

To communicate with smart contract agents and blockchain agents, human agents must own a digital wallet to interact with smart contract agents. The existence of a wallet agent replaces the banker who handles all the transactions in the lottery system. A blockchain

agent is a distributed ledger that holds all the transaction details made by a human agent. The data on the blockchain agent is immutable and secure. Web agent is a simple platform to ease human agents to access and interact on the same version and features that support the systemization of information. The player agent gets the latest information and buys tickets through the web agent.

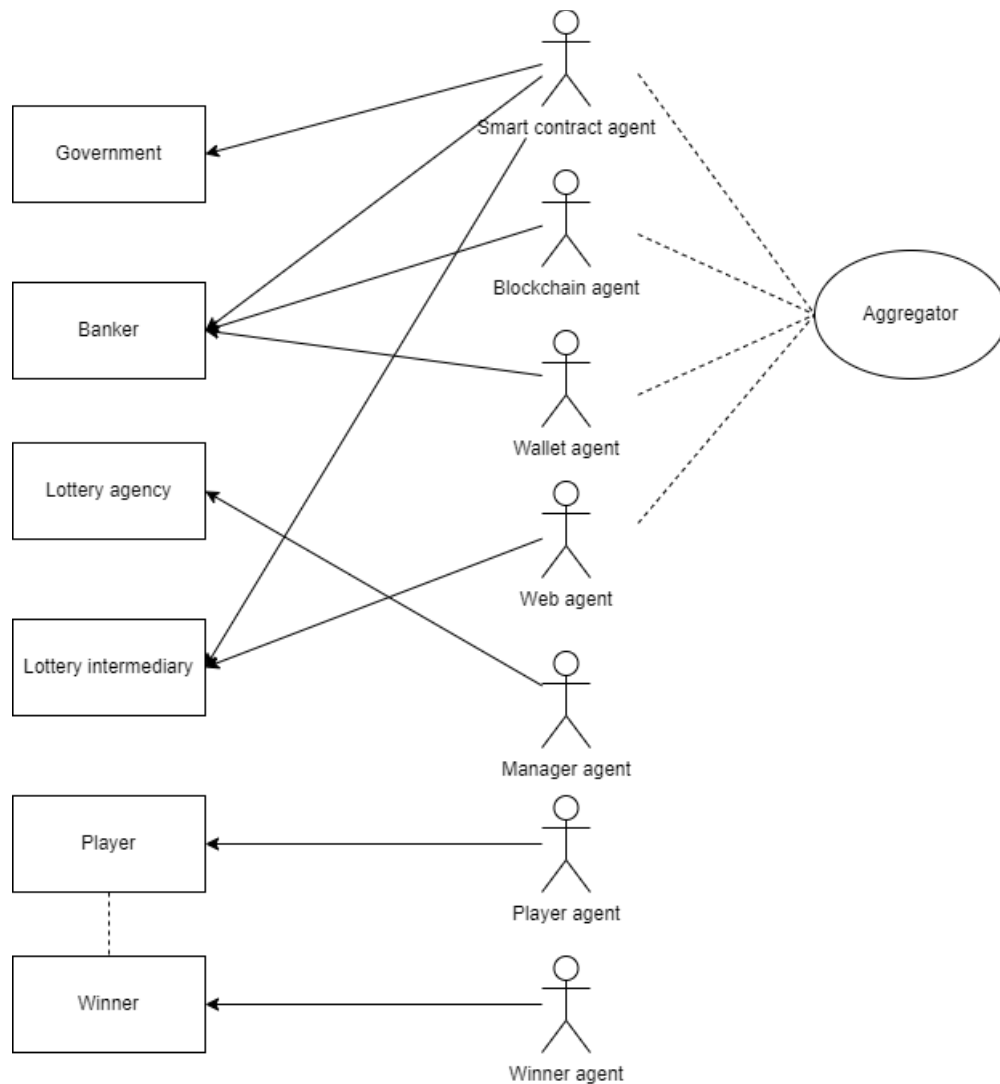


Figure 4.8: Role Mapping of the Actors and Agents

4.3.4 Phase 4: Platform Independent Design and Modelling

In the previous description, the blockchain requirements have been identified. The design of the blockchain application is presented in this phase. This involves the modelling of the knowledge model, interaction model, scenario model, and behaviour model.

Interaction Modelling

As shown in Figure 4.8, manager agent, player agent, blockchain agent, smart contract agent, wallet agent, and web agent are all part of this system. Therefore, the interaction model shows how the agents communicate to exchange information until they achieve the goal. This interaction model shows the human agent (player agent or manager agent) who attempts to enforce the smart contract.

Figure 4.9 shows that the player agent starts by reading the rules. Before joining the game, the player agent must agree on the rules and enter the desired amount to stake. Once the player agent joins the game, the wallet agent is triggered to request the player agent's signature to proceed. After the player agent confirms it, the smart contract agent is enforced based on the conditions. The block is then mined on the Blockchain and returns the transaction details; then, the player agent is redirected to the success page. By then, the payment of the bet is completed.

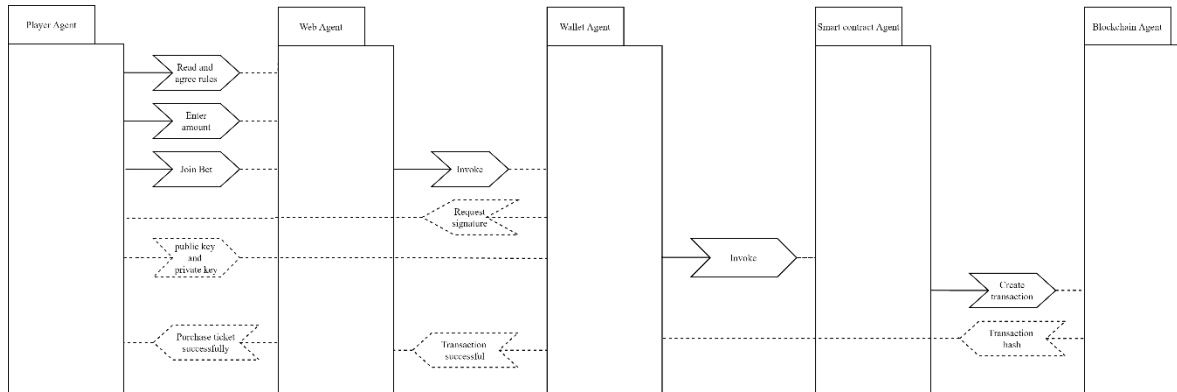


Figure 4.9: Interaction Model When Player Join Bet in Blockchain-based Lottery Application

Knowledge Modelling

Knowledge models in Figure 4.10 represent the private and shared knowledge among the agents. The human agents (manager, player, and winner) have their wallet details, such as public key, private key, and account balance, which should be kept private. Human agents participating in this blockchain-based lottery 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 portion of prize distribution, and the random number algorithm. Once the players join the game, the player address is added to the ticket array and visible among the players and the manager. The report in the smart contract is kept private, containing the contract balance from the player and the amount of prize and commission. All the transaction made is mined into a block on the Blockchain, which is public. Lastly, the result will only be revealed to the winner.

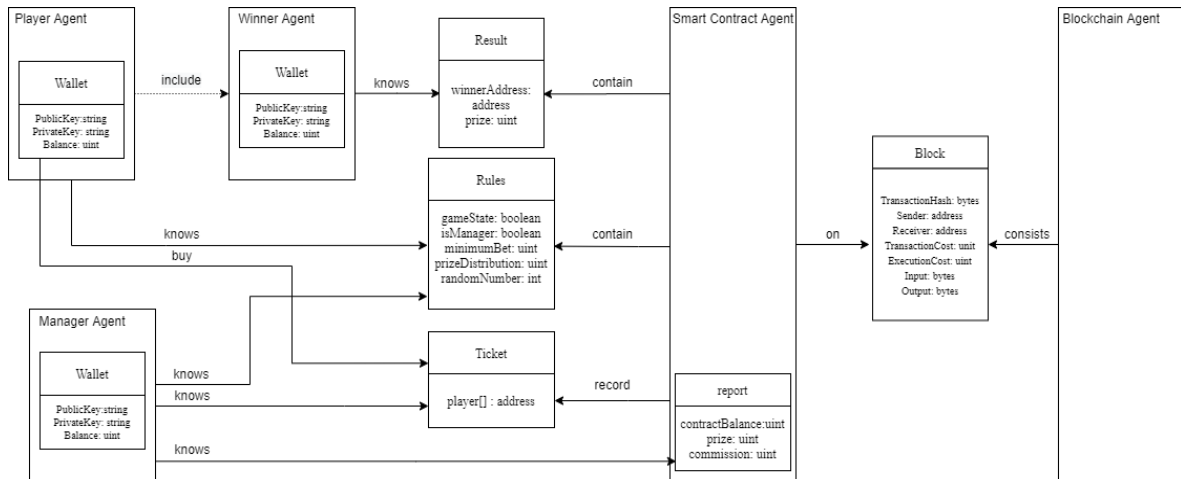


Figure 4.10: Knowledge Model for Blockchain-based Lottery Application

Scenario Modelling

The scenario model shows the sequence of steps on how the agent starts and achieves the goal. In scenario 1, the human agent player agent aims to perform a transaction to buy a lottery. Since the application adopts a blockchain application, the quality goal of this scenario is to replace the middleman using a smart contract. At the end of the activity, the data transaction of the player agent will be stored on the blockchain.

Table 4.7: Scenario Model 1 for Ethereum Blockchain-based Lottery Application

Scenario 1				
Quality Goal		Remove central authority, transparency		
Initiator		Player Agent		
Trigger		Store transaction detail on Blockchain		
Description				
Condition	Step	Activity	Agent types/role	Immutable/mutable resources
	1	Access the web application	Player agent	Web application - mutable
	2	Check the rules and regulations	Player agent	Smart contract - immutable

Table 4.7 continued

	3	Agree on the rules and regulations	Player agent	-
The game state is open	4	Join game	Player agent	-
	5	Connect wallet	Player agent	-
	6	Set bet amount	Player agent	Bet amount - mutable
Enough balance	7	Confirmed transaction	Player agent	-
Provide signature	8	Enforced smart contract	Web3 agent	Smart contract - immutable
Passed precondition	9	Mine a block	Miner agent	-
	10	Transaction stored in the block	Miner agent	-
	11	Update contract balance	Blockchain agent	-
Transaction successful	12	Return hash	Smart contract agent	Blockchain - immutable

In the second situation, the intermediary is also eliminated. The smart contract is responsible for secure and efficient management. Smart contracts, on the other hand, aid in the tracing of data stored in the blockchain.

Table 4.8: Scenario Model 2 for Ethereum Blockchain-based Lottery Application

Scenario 2	
Quality Goal	Remove central authority
Initiator	Manager Agent
Trigger	Trace blockchain data

Table 4.8 continued

Description				
Condition	Step	Activity	Agent types/role	Immutable/mutable resources
	1	Visit manager portal	Manager agent	Web application - mutable
	2	A smart contract is triggered	Web3 agent	Web application - mutable
Is manager	3	Read data from Blockchain	Smart contract agent	Blockchain - immutable
	4	Get the balance amount	Smart contract agent	Blockchain - immutable
	5	Display the amount	Web agent	Blockchain - immutable

Scenario 3 shows the integrity while using blockchain applications. The manager can efficiently distribute prizes by executing the smart contract. In this case, smart contracts could save time and eliminate human error or mistakes, such as miscounting the bet amount.

Table 4.9: Scenario Model 3 for Ethereum Blockchain-based Lottery Application

Scenario 3				
Quality Goal		Integrity		
Initiator		Manager Agent		
Trigger		Distribute prize and commission		
Description				
Condition	Step	Activity	Agent types/role	Immutable/mutable resources
	1	Visit manager portal	Manager agent	-
	2	Select winner	Manager agent	Web application - mutable

Table 4.9 continued

	3	Connect wallet	Manager agent	-
	4	Pay gas	Manager agent	Wallet balance - mutable
	5	Confirmed transaction	Manager agent	-
Provide signature	6	Trigger smart contract	Web 3 agent	-
Is manager	7	Read data	Smart contract agent	Blockchain - immutable
The player list is not empty	8	Loop player address	Smart contract agent	Blockchain - immutable
	9	Random select one winner	Smart contract agent	-
	10	Get contract balance	Smart contract agent	Blockchain - immutable
	11	Calculate commission	Smart contract agent	-
	12	Calculate prize	Smart contract agent	-
	13	Mine a block	Miner agent	Blockchain - immutable
	14	Block created	Miner agent	Blockchain - immutable
	15	Distribute commission to manager	Smart contract agent	-
	16	Distribute prize to the winner	Smart contract agent	-
	17	Return hash	Smart contract agent	Blockchain - immutable

Behaviour Modelling

Figure 4.11 shows the behaviour of a player participating in the game. First, the player must visit the blockchain-based lottery application and agree to follow the rules and regulations. Then, the wallet is triggered automatically to request a signature. The transaction triggers the smart contract and performs condition checking. Firstly, it checks the game state, then the address of the player, and the bet amount. The transaction is rejected if the player's address belongs to the manager or did not achieve the condition. After passing the condition, the prize pool is updated, and the player address is added to the array list as a ticket placement. Then the player is notified by receiving a hash.

Figure 4.12 displays the manager's behaviour when selecting a game-winner. The manager visits the web application and checks on the game's state. If the game state is true, the web application displays the contract balance, referring to the smart contract. Then, the manager selects a winner on the web application; the wallet is triggered to request for signature. The smart contract is triggered and proceeds with the following steps. First, the smart contract updates the game state to false to halt the game. The smart contract ensures the manager performs the transaction and the player address array is not empty. Next, the smart contract populates a random number that represents the array index of the player list. The prize and commission are calculated and transferred to the winner and manager. Lastly, the smart contract clears the player list and updates the game state to true to be ready for the next game. The winner and manager will be notified by receiving the hash.

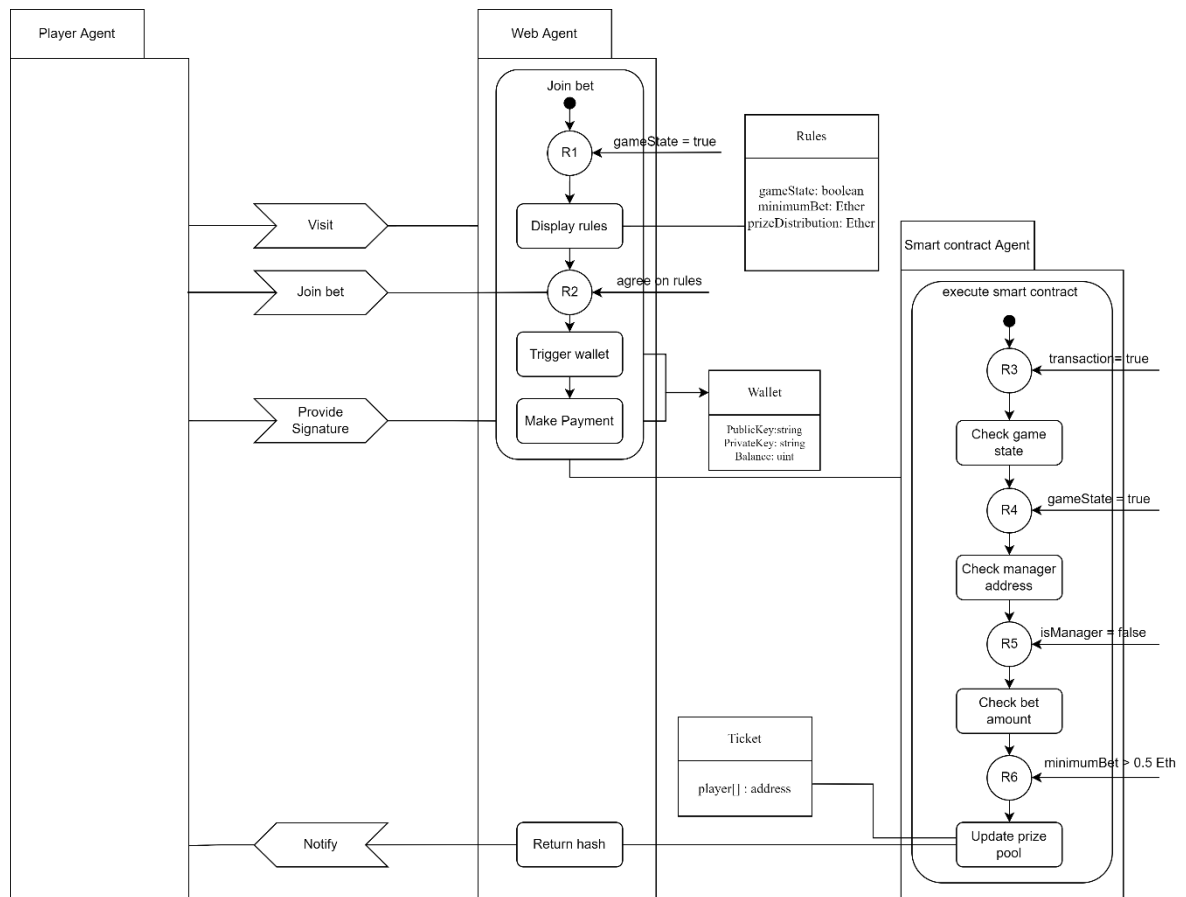


Figure 4.11: Behaviour Model for Player Join a Bet in Blockchain-based Lottery Application

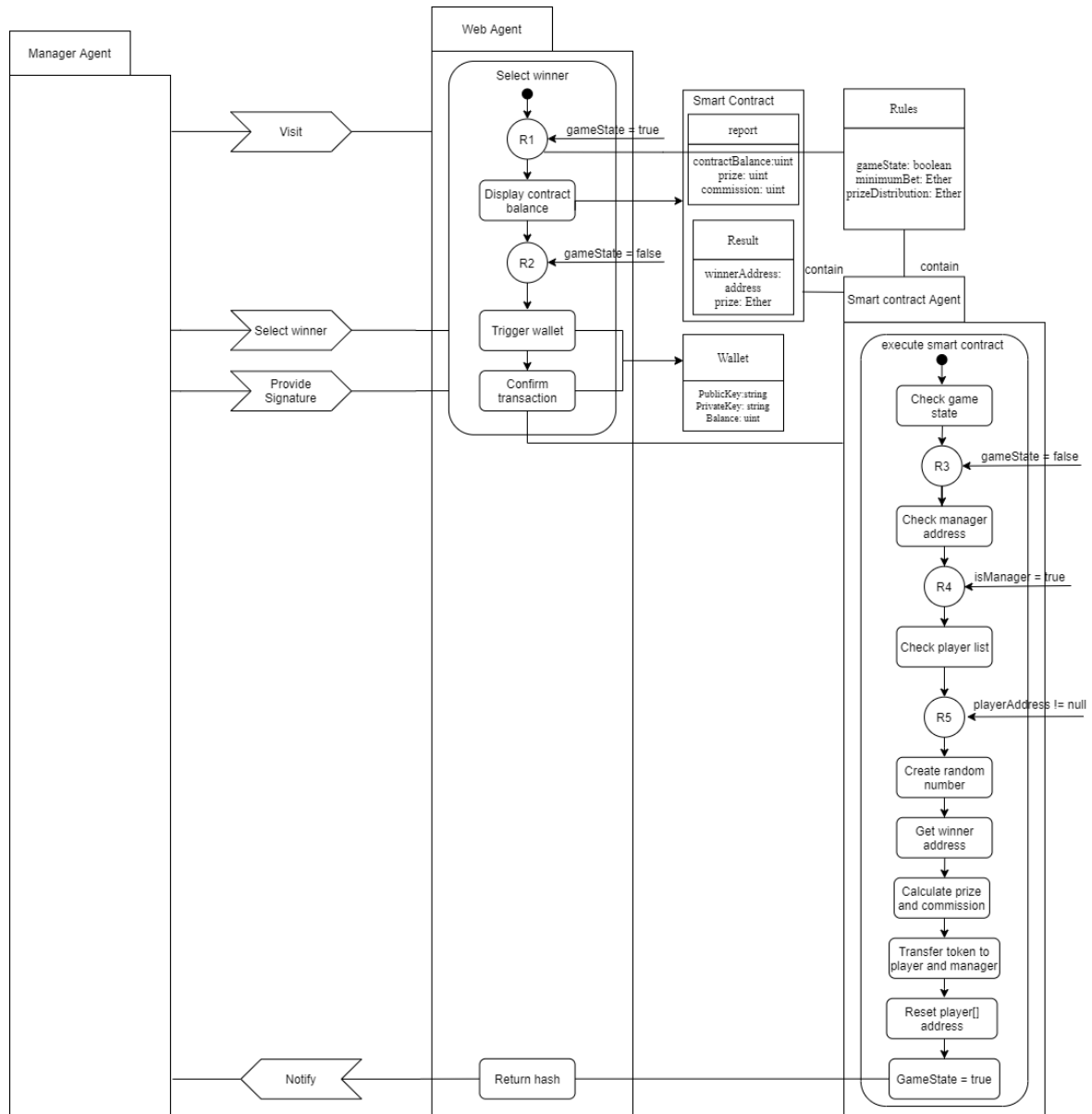


Figure 4.12: Behaviour Model for Manager Select a Winner in Blockchain-based Lottery Application

4.3.5 Phase 5: Platform Specific Model

The transformation rules in Chapter 3 are adopted to transform the platform-independent model into a platform-specific model in this phase. Thus, we start to map the knowledge item in behaviour models into code.

Table 4.10: The Transformation Guidelines for Blockchain-based Lottery Application

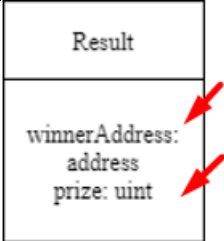
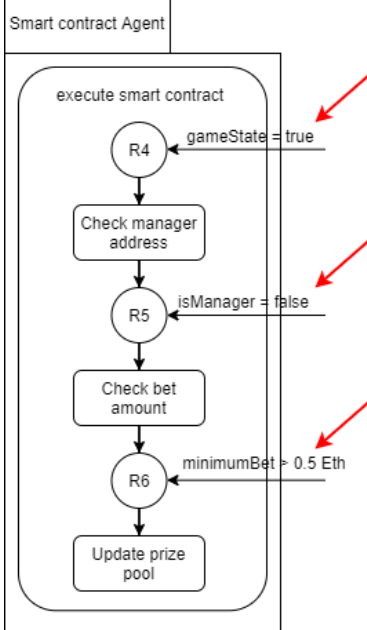
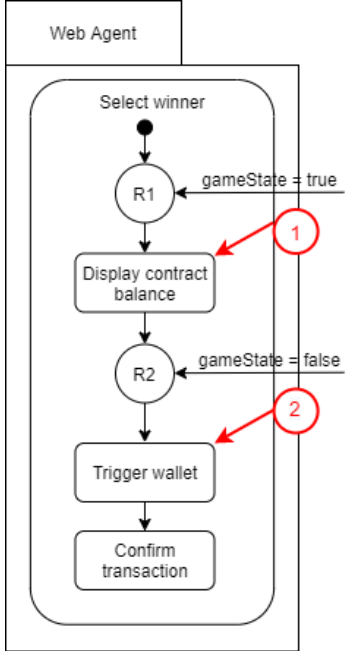
Model Artifacts	Corresponding Code	Samples
	Contains smart contract attributes or property	<pre>uint prize; address winnerAddress;</pre>
	Contains the smart contract conditions	<pre>function sendEther() { require(gameState == true); require(msg.sender != manager); require(minimumBet > 0.5 ether) //then update prize pool }</pre>

Table 4.10 continued

	Contains the action in the web agent that handle in javascript	1. Display contract balance <pre>render: async () => { // Load the balance from the blockchain const balance = await App.lottery.get_balance() }</pre> 2. Trigger wallet <pre>loadAccount: async () => { App.account = web3.eth.accounts[0] },</pre>
--	---	---

Development Phase

Once the model is designed, the developer moves into the implementation phase based on the modelling notations. Developers can be divided into two teams to develop blockchain-based lottery applications. The first team can handle the smart contract development, and another team will develop the front end using HTML, CSS, and web3. Once both teams have done their parts, the web application can connect to the smart contract using Web3. Soon after, the web application can test locally to ensure no errors or bugs. Lastly, the application is ready to be deployed.

Blockchain-based Lottery Application (Screenshot)

Ethereum Based Lottery System 0xce06030501da2dea522504d499c33af6da1fd4b8

Welcome to lottery system

01.01.2021 22:19

Powered by Ethereum Blockchain

A worldwide Ethereum-based Lottery System.

Introduce a lottery game system with high security, but low cost.
Join us now!!

read and check on the rules, enter an amount and hit enter.

Ether Value...

☐ [Read the rules and agreement](#)

Figure 4.13: The Main Page of the Lottery Application

Ethereum Based Lottery System

Rules and Regulations

1. Lottery-Based System is powered by Ethereum Blockchain.
2. Every round of the game, each player can only join the game **ONCE**.
3. The **MINIMUM** amount to join the game is 0.01 Ether.
4. Only **ONE** winner will be selected.
5. Game will be restart on every Saturday.
6. **10%** of the prizes will be draw by the manager as comission.
7. Prize will be reveal on every **FRIDAY**.
8. Once transaction made, the transaction cannot be alter..

** gas are not included in the bet amount.*

[Back](#)

Figure 4.14: The User Interface of Rules and Regulations

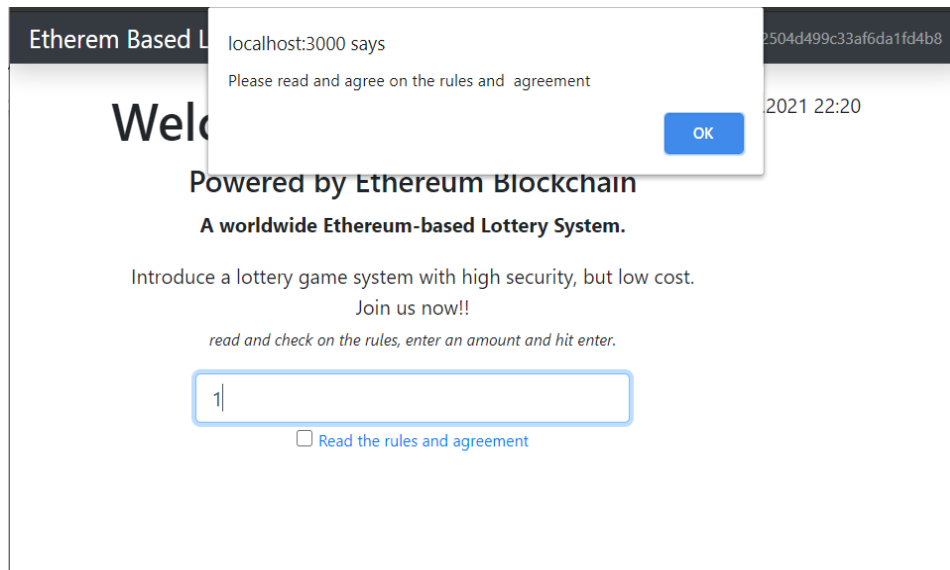


Figure 4.15: Prompt Message When Player Does Not Checks on the Agreement

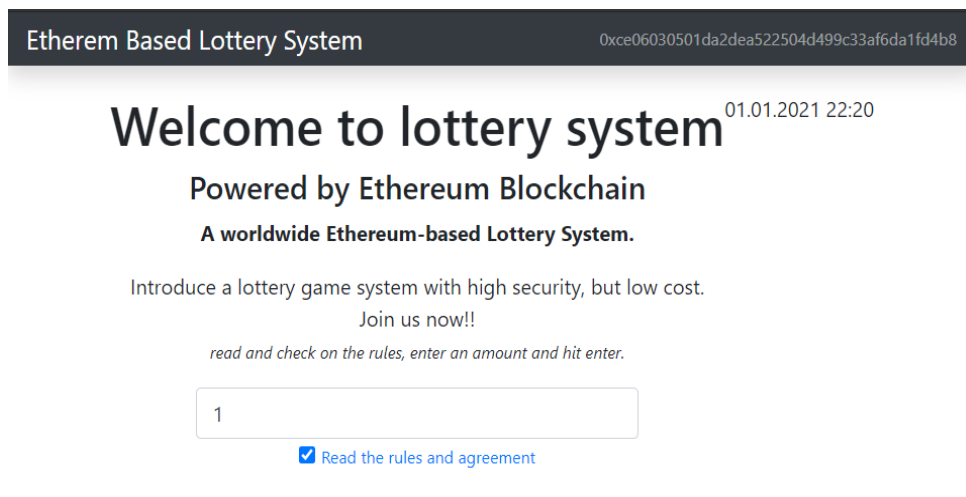


Figure 4.16: When Player Enters the Amount and Checked on the Agreement

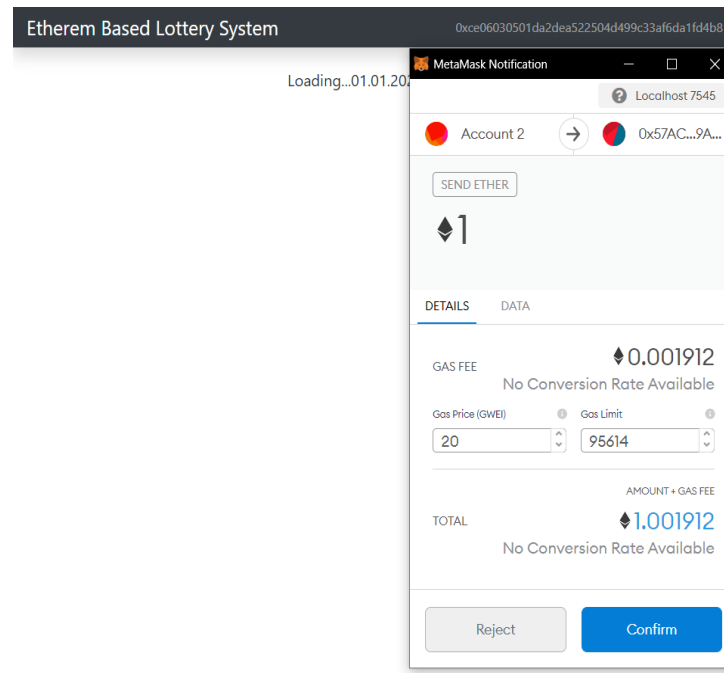


Figure 4.17: Display Metamask After Player Joins the Bet

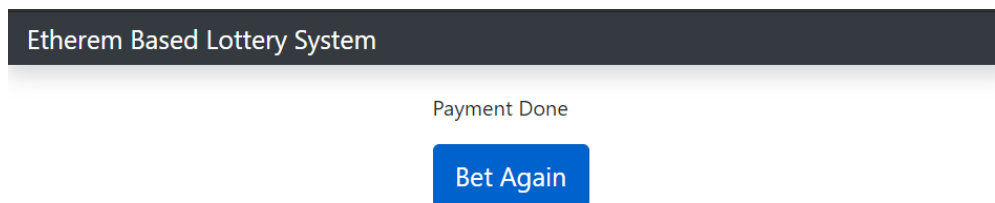


Figure 4.18: Redirect Player to the Success Page Once Transaction Done

4.4 Lesson Learnt

This section illustrates the lesson learnt from the developing the blockchain-based lottery application through extended AOM. The case study leads to the following lessons:

1. With extended AOM, this case study has demonstrated a systematic approach to developing a blockchain-based lottery application. The methodology entails breaking down a difficult problem into layers such as HOMER, computation independent modelling, early identification of blockchain use cases, platform-

independent modelling, and platform-specific modelling. The requirements are elicited through a technique known as HOMER by reviewing the recent literature review. In the computation independent modelling, elicitation answers are mapped into a goal model, role model, organisation model, and domain model. Then, the e^3 value model will be used to identify the potential of the blockchain. Next, the blockchain properties will be designed in the platform-independent modelling layer through the knowledge model, interaction model, scenario model, and behaviour model. Lastly, the platform-specific modelling will translate the model into code based on the transformation rules.

2. With the extended AOM, the technical and non-technical clients will have a clear goal for the project. The methodology will emphasise “*what to solve*” and “*how to solve*”. The HOMER can lead the stakeholders to think critically about “*what to solve*” for a use case. Then the modeller will map these requirements in the computation independent modelling. Then, the modeller can identify the potential of the blockchain use case by using the e^3 value model. Lastly, the modeller can continue designing the blockchain application on the platform-independent modelling and answer “*how to solve*” problems using blockchain technology.
3. Compared to the current software methodologies for blockchain application development, as stated in Chapter 2, the extended AOM offers analysis and design support. Using extended AOM, the blockchain potential could be identified by using the e^3 value model before diving deep into the design phase. This also provides a clearer view of which intermediary could be replaced by using the blockchain to optimise the benefits and usage of blockchain technology.

4.5 Conclusion

This chapter discusses the lottery case study using the extended AOM. It covers the requirement elicitation using HOMER, then constructing the computation independent modelling, the identifying the blockchain use case using e^3 value model, and platform-independent design. Next, the model is converted into codes based on the transformation rules stated in platform-specific modelling. Finally, the lessons learned from the case study are discussed in this chapter. The next chapter presents a quantitative and qualitative evaluation of the extended AOM.

CHAPTER 5

EVALUATION OF THE EXTENDED AOM

5.1 Overview

This chapter presents the evaluation of the extended AOM. The evaluation aimed to investigate the needs of AOM and extended AOM for blockchain application development among novices. In addition, the evaluation is conducted to study the usability of extended AOM. Section 5.2 presents the experiment settings and analysis methods. Section 5.3 covers the details of experiment 1. Experiment 1 presents the original AOM and UML adoption in blockchain application modelling. The aim of experiment 1 was to demonstrate the usability of AOM in blockchain application modelling in line with the interest in using UML in blockchain application modelling. Section 5.4 covers the details of experiment 2. Experiment 2 studied the limitations of AOM in blockchain application modelling. The aim of Experiment 2 served as the rationale for the use of extended AOM. Section 5.5 covers the details of experiment 3. Experiment 3 demonstrated the usability of the extended AOM in blockchain application modelling. Section 5.6 presents the findings based on the experiments. Section 5.7 illustrates the threats to the validity of the experiments. The chapter is concluded in Section 5.8.

5.2 Experiment Setting

Quantitative analysis is adopted to understand the behaviour of the modelling techniques (UML, AOM, and extended AOM) when modelling blockchain applications. This research conducted three experiments to understand novice behaviour when modelling blockchain applications. Experiment 1 presents the original AOM and UML adoption in blockchain application modelling. The aim of experiment 1 is to demonstrate the usability

of AOM in blockchain application modelling in line with the interest in using UML in blockchain application modelling. In experiment 1, the participants were first required to model the blockchain requirements through UML and followed by AOM based on the case study in Chapter 4. A scoring system is derived and designed from the blockchain properties, as mentioned in Table 5.1 and the case study in Section 5.3.1 and Section 5.4.1. Then, scores were given according to the scoring system in reference to the participants' answers.

Table 5.1 presents the definition of blockchain properties. The scoring system presents the comprehensiveness of blockchain technology through the models. A higher score means the models can explicitly capture blockchain needs. A higher score indicates that the models can capture the blockchain requirements explicitly. The scoring system included the blockchain properties involved in the case study. Each of the properties included in the model gains a point.

Experiment 2 examined the limitation of AOM in modelling a blockchain application. The aim of Experiment 2 served to provide the rationale for supporting the need for an extended AOM. As AOM presents a different level of abstraction in complex socio-technical system modelling, we believe that participants were able to present the blockchain development in a clear-cut manner. In experiment 2, participants were required to identify the problems stated in the case study. They must analyse and model the problem through AOM in computation independent modelling. Then, they were told to design the blockchain application in platform-independent modelling phase.

Experiment 3 examined the usage of extended AOM to model a blockchain application. Experiment 3 asked participants to model the case study through extended AOM. Then, a scoring system was applied to the models. Finally, a usability study is

conducted to examine the benefits of the extended AOM for blockchain application development.

Experiments 2 and 3 conducted the same scoring mechanism as Experiment 1 but with a different scoring schema. The students accumulate the score based on the respective schema. The higher score indicates that the models can explicitly capture blockchain requirements.

Table 5.1: Definition of the Blockchain Properties

<i>Blockchain properties</i>	<i>Definition</i>
<i>Cryptocurrency</i>	A replacement of the traditional currency to perform a transaction (Farnoush, Gupta, Ahady, & David, 2021)
<i>Authentication</i>	Confirming the identity of the requestor to access the data (Choi & Ouyang, 2021).
<i>Data integrity</i>	The capacity to detect the altered data (Farnoush et al., 2021).
<i>Transparency</i>	Allows all logic and information in the Blockchain visible (Dinh et al., 2018).
<i>Trustworthy</i>	The Blockchain's transparency value serves as the cornerstone for the trustworthy. Users do not have to trust each other due to the stated rules in a smart contract, as their action is predefined by the set circumstances (Nzuva, 2019).
<i>Immutability</i>	Ability to withstand any data modification (Farnoush et al., 2021).
<i>Autonomous</i>	The logic and the rules stated in the smart contract can run automatically (Li & Fang, 2021).
<i>Fairness</i>	The smart contract serves the fairness properties to guarantee on some condition by predefined rules.

Generally, the experiments were designed to test the following hypotheses by comparing numeric variables by using means:

H1: *The number of derived blockchain requirements in AOM is higher than in UML.*

As Rocha and Ducasse (2018) mentioned, UML is unable adequately capture the specification behaviour of blockchain applications. Thus, AOM can derive more blockchain requirements than UML because AOM can handle technical systems.

$$H_1: \mu_{AOM} > \mu_{UML} \quad \text{Equation 5.1}$$

H2: *Extended AOM can help novices design a blockchain application with hidden properties to solve the problem.*

The extended AOM provided a complete process stage starting from analysing the problem, identifying the needs of the blockchain, and designing the blockchain elements to solve the problems. These processes help to guide the novice to exert their blockchain knowledge to a solution.

$$H_2: \mu_{\text{extended AOM}} > \mu_{AOM} \quad \text{Equation 5.2}$$

5.2.1 Students' Background

The experiment was conducted on two batches of Bachelor of Computer Science (Software Engineering) students from the Universiti Malaysia Sarawak who enrolled for TME4093, Advanced topic in Software Engineering. The first batch of students is from Semester 2, 2019/2020. A total of 90 students participated in Experiment 1.

Batch two's participants comprised 140 students enrolled in TME4093, an Advanced topic in Software Engineering in 2021. Experiments 2 and 3 were conducted among batch

two students to understand the limitation of AOM and the usefulness of extended AOM in modelling blockchain applications.

The first batch of students underwent Experiment 1, which served as a pilot study to evaluate the methods and procedures that would be implemented on a bigger scale. The experiment was then modified and tested on the second batch of students for further analysis.

Technically, the participants are novices and have zero knowledge of Blockchain since blockchain technology has yet to be covered in the syllabus. However, the students know UML and AOM as they learned UML in TME2413, Object-Oriented Software Development, during their second year and AOM in TME4093, Advanced Topics in Software Engineering, in their fourth year.

5.3 Experiment 1 – The Usage of UML and AOM for Blockchain Application Development

As mentioned before, the aim of Experiment 1 is to evaluate the usage of UML and AOM in eliciting blockchain requirements. The findings from this experiment will support hypothesis 1. It can also be used as a pilot study to understand the importance of software modelling in blockchain application development.

For the first stage of the experiment, the students were given a task to model the blockchain application using UML. In the second stage, the students were required to model the case study through AOM. Then, the models were evaluated based on a scoring system.

5.3.1 The Walkthrough of the Lottery Case Study for Experiment 1

A walkthrough case study is elaborated on in this section. The walkthrough case study was derived based on the lottery blockchain games (Liao & Wang, 2017; Saichua et al., 2019). In the case study, the detailed requirements of a lottery system were derived

together with blockchain requirements. The blockchain requirements were derived from (Farnoush et al., 2021), (Choi & Ouyang, 2021), (Dinh et al., 2018), (Nzuva, 2019), and (Li & Fang, 2021). The requirements are highlighted in Table 5.2. The case study description is as follows.

A lottery is a type of game to provide players with the opportunity to win big prizes. Some people odds to play the lottery in the hope of winning a big fortune; some are for fun. However, there is a trust issue between the lottery industry and the player. To win a fortune, many people would wonder if the deal and ticket for the winner are real. There is also a concern if the winning prize would be given on time or in the amount promised. Therefore, blockchain technology can solve the trust issue. By adopting blockchain technology to the lottery industry, players will buy with confidence since most smart contracts are public. Players can **check all the rules and agreements before the bet**.

The lottery's smart contract will be implemented using the Ethereum platform. There are two main parties to operate the lottery's 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. The lottery game starts 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 paying a **minimum of 0.01 Ethers** but without the maximum limit to try the gameplay. Each player who enters the game is **added to an array list**, and the **contract holds the amount of Ethers** sent by the players. The **manager can check the total amount of Ethers** from all the players. Once the time is up, the lottery game manager will **execute the contract to select a winner**. In the contract, **a random number is first generated** from the array list, and a winner is selected. Lastly, the total amount of **Ethers in the contract is automatically**

transferred to the winner's wallet. Therefore, the prizes won by the winner are assured. Then the **array list is reset** to prepare for the next game. The requirements in the case study are listed in Table 5.2.

Table 5.2: Requirements Based On The Case Study

No.	As a ...	Will be able to ...	Leads to ...
1	Player	Check the rules and regulations before bet	Transparency
2	Player	Send ether through wallet	Cryptocurrency
3	Smart contract	Check the minimum ether	Autonomous
4	Smart contract	Handle a list of player address	Immutability
5	Smart contract	Handle ether sent by the player	Immutability
6	Manager	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

5.3.2 Data Analysis for Experiment 1

Figure 5.1 presents the findings on modelling blockchain applications through UML and AOM. The Y-axis represents the number of students, while the X-axis shows the number of scores achieved. The chart shows that the highest score to achieve in both modelling techniques is 9/10. Interestingly, the percentage of the students achieving 9/10 in AOM is

23.34% higher than UML (26.67% and 3.33%, respectively). The mean score for UML and AOM is tabulated in Table 5.3. There is a slightly different mean score in using UML and AOM for blockchain modelling. Hence, the H1 hypothesis was proven. However, both UML and AOM are able to model blockchain requirements explicitly. Although the students are new to AOM, the AOM can model a complex socio-technical system like blockchain just as well as UML. From the review, lots of works adopted UML as a modelling language for blockchain applications (Jurgelaitis, 2019; Rocha & Ducasse, 2018b). Experiment 1 can lead to the claim that AOM can serve as an alternative modelling language to model a blockchain application.

A survey was conducted to understand the students' preference for UML or AOM for blockchain application development, and the result is shown in Figure 5.2. It is interesting to report that 10% of the students strongly agreed that AOM could reflect on blockchain application development; 40% agreed that AOM is useful due to the completeness of the models and ease of understanding. On the other hand, 33.33% of the students rated neutral as most lack blockchain technology knowledge; 16.67% disagreed that AOM is useful for blockchain application development and claimed that UML is more suitable to model blockchain applications.

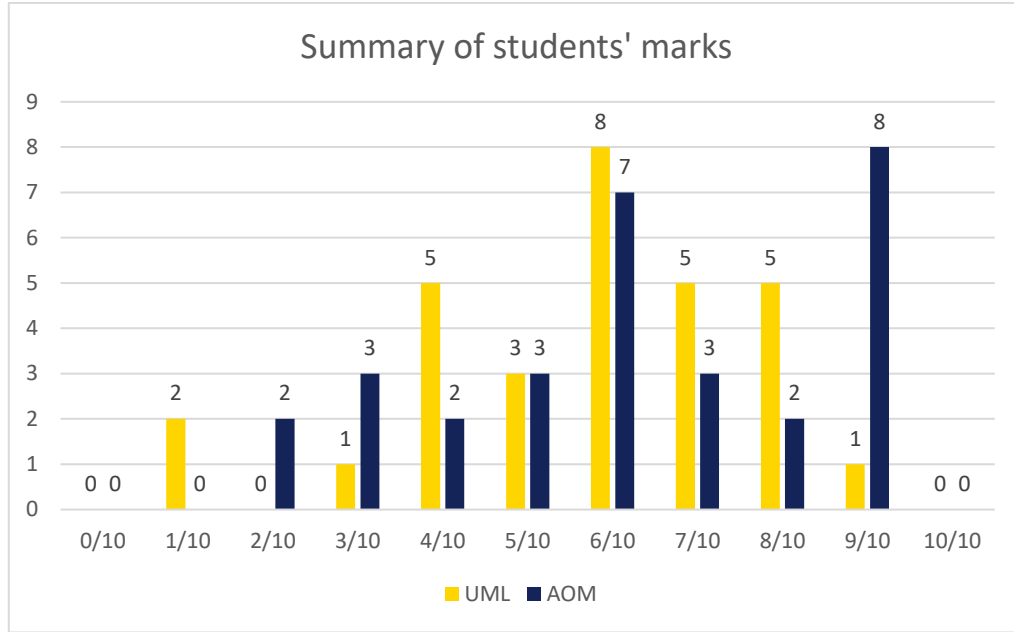


Figure 5.1: The Summary of Students' Marks

Table 5.3: Mean Score for UML and AOM

<i>Modelling Technique</i>	<i>UML</i>	<i>AOM</i>
<i>Mean</i>	15.636	17.000

In sum, AOM can serve as an alternative modelling language for blockchain-based application development. Although it can model the blockchain requirements, many students still failed to capture the blockchain requirements based on Experiment 1. Hence, the limitation of AOM in blockchain application development that led to the issue was further investigated in Experiment 2.

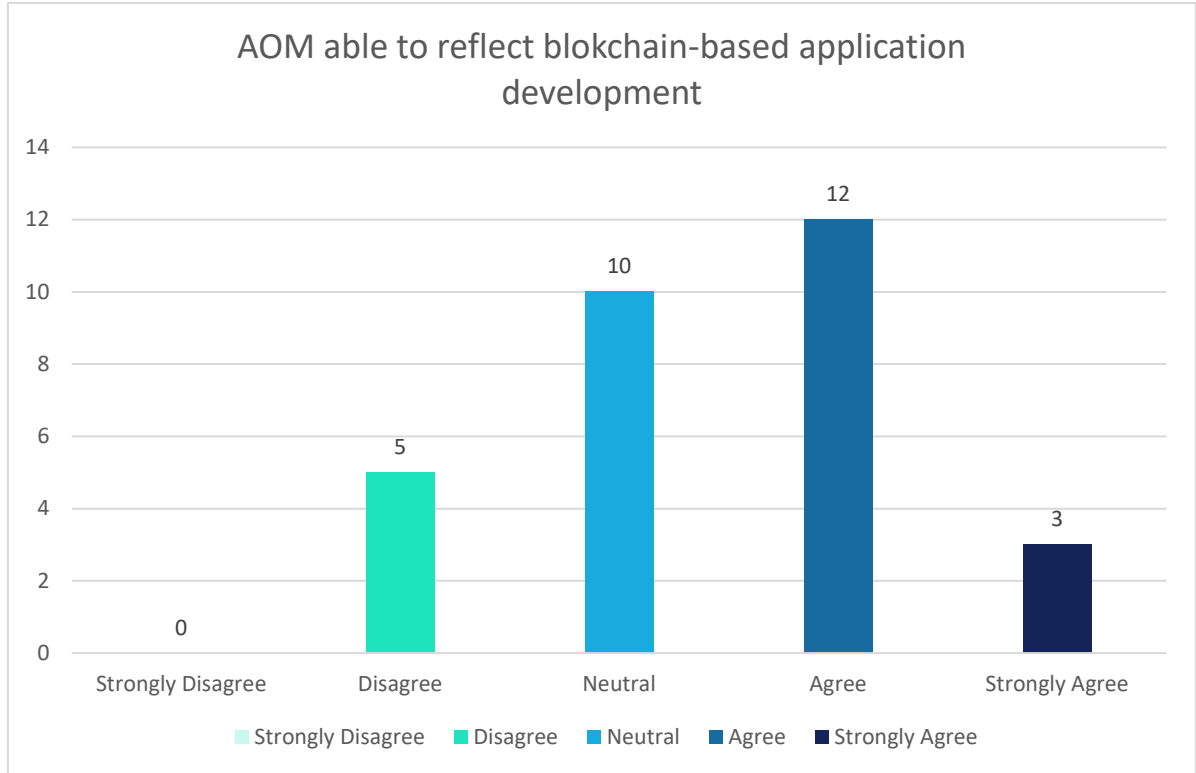


Figure 5.2: Survey Result Analysis

5.4 Experiment 2 – The Usage of AOM for Blockchain Application Development

The aim of Experiment 2 is to investigate the limitations of AOM as a modelling language for blockchain-based application development. As AOM has introduced a different abstraction layer for the system, a detailed analysis will be conducted to study how the abstraction can reflect blockchain-based application development among students. It is important to evaluate whether the students can understand the abstraction layer of the AOM and how it is related to blockchain application development.

Based on the pilot study, we refined the case study on the lottery industry. In Experiment 1, the case study focuses on the intensive study of a blockchain solution for the lottery industry by providing blockchain properties. The second experiment described the

industry's dilemma and was designed to allow the novices to study the potential of blockchain for the lottery sector, and the technology's impact.

From the case study, we expected the students to derive the blockchain requirements models in the experiment by using AOM and extended AOM. In this case, the models should be able to present how blockchain replace the intermediaries, how player access the smart contract to develop a trusted environment, how the smart contract executes the logic and the predefined rules to manage fairness, how the transaction was made by the player and how the "global ledger" is utilized to achieve data integrity. In addition, we further investigate whether those software models can capture the blockchain requirements and identify any hidden requirements through the modelling process.

5.4.1 The Walkthrough of the Lottery Case Study for Experiment 2 and Experiment 3

The case study is known as the lottery game, which provides an opportunity to win a big prize. The following elaborates on how Alice buys a lottery ticket, waits for the ticket draw and goes through the redemption process.

Alice experiences her first lottery ticket purchase. She purchases the lottery ticket by visiting a physical lottery store. The lottery intermediary explains the rules and regulations to Alice and encourages her to buy a ticket. Then Alice buys the ticket and waits for the lottery to be drawn. During the drawing process, Alice can see the result from social media such as newspapers, TV, and radio. The lottery agency randomly draws a number that represents the ticket number as a winner. At that time, Alice won the lottery and went to the physical store to claim her reward. During the redemption process, Alice encountered some issues.

1. The official lottery had some things that needed fixing. It was hacked and corrupted. Alice failed to redeem the prize.
2. Alice bought the ticket through a lottery intermediary without visiting the physical store. After winning, the intermediary swallowed the prize and ran away. Alice failed to redeem the prize.
3. Alice bought a lottery ticket by visiting the physical store. After winning, the lottery company claimed that her ticket was fake. Alice failed to redeem the prize.
4. Before redeeming the prize, Alice accidentally lost her lottery ticket. Alice failed to redeem the prize.

From the elaboration, the lottery industry has much room for growth. One of the industry's most significant problems is that the players lose faith in it. There needs to be more transparency, and there is increasing doubt about the randomness of lottery draws and business audits. Besides, the players are concerned about the validity of the prizes, and most of the winners prefer Lump Sum Payments to get all the money immediately.

From the lottery industry perspective, the lottery ticket sales process requires a large workforce, leading to high labour costs. On the other hand, the overall productivity of the lottery ticket distribution operation could be better. The industries must deal with scalability while the lottery market is expanding.

Throughout the process, some intermediaries are managed by the government. Intermediaries sell the lottery to the citizens in each area. However, the government sets the lottery ticket price and sells at a high rate, but the reward to the winner is lower than the desired rate. However, some intermediaries market the lottery ticket at a higher price than

the government's set price, a kind of corruption that includes other types of fraud. Thus, the reliability and integrity of the lottery industry are threatened.

Due to the low threshold for lottery sales, many sites, and many lottery intermediaries, it is not easy to prove the validity of the lottery ticket and the winner. The traditional lottery did not perform real-name authentication, which resulted in the loss of lottery tickets and the tickets used by others. Sold tickets are difficult to trace, whether the tickets are sold from an official platform.

When the lottery market is expanding, a third party, the banker, will handle the company's financial flow. The banker is responsible for providing financial solutions, managing the monthly transaction, and generating a monthly financial report for the lottery company. However, the company might need to question the trustworthiness of the third party to handle the entire financial management.

Hence, blockchain technology is introduced to overcome trust issues, transparency on rules and regulations, ticket authentication, trace tickets, trace sales, fairness in getting a random number, and fast settlement on the reward. To enhance the game's fairness, only non-lottery agencies can join the game, and each player can only join once.

The use case highlighted several issues: fairness of the game when drawing the winner, transparency of the game rules, scalability when the market expanded, and the existence of middlemen such as bankers and intermediaries. Based on the case study, we have identified the blockchain requirements in Table 5.5 based on the case study. As mentioned, the criteria captured in this experiment are fairness, transparency, cryptocurrency, immutability, and data integrity of blockchain technology. Additionally, the criteria are based on the Blockchain and smart contract properties mentioned by Jairam,

Gordijn, Da Silva Torres, Kaya, and Makkes (2021) and Farnoush, Gupta, Ahady, and David (2021). Jairam et al. (2021) mentioned that a fair distribution of government power should be considered a tech platform. Governance entails establishing rules and on the other, constantly monitoring whether systems adhere to the specified rules.

5.4.2 Scoring System

As mentioned in Experiment 1, a scoring system covers the quality of the blockchain property in the models. The scoring criteria for experiment 2 are stated in Table 5.4. It was designed based on the abstraction layer from AOM. Each of the criteria holds an equal weight.

The computation independent modelling aimed to let the students analyze the problem from the human perspective. The platform-independent modelling covers blockchain properties that will be designed in the application. We excluded the deployment layer in this experiment due to the limited time in the experiment.

Table 5.4: The Blockchain Requirements for the Lottery Case Study

Computation independent modelling		
No	Criteria	Point
1.	Is the model at this layer able to show fairness issues?	1
2.	Is the model at this layer able to show transparency issues?	1
3.	Is the model at this layer able to show scalability issues?	1
4.	Is the model at this layer able to show how bankers manage the issue?	1
5.	Is the model at this layer able to show how intermediaries sell a ticket?	1
Platform independent modelling		
No	Criteria	Point

Table 5.4 continued

1.	Is the model at this layer able to show how smart contracts solve fairness?	1
2.	Is the model at this layer able to show how Blockchain provide transparency?	1
3.	Is the model at this layer able to show the transaction made through a smart contract?	1
4.	Is the model at this layer able to show how smart contracts arrange the player ticket?	1
5.	Is the model at this layer able to show how data is store on the Blockchain?	1

Experiment 2 was conducted among 140 students enrolled in Semester 2 2021 for the course TME4093, Advanced Topic in Software Engineering.

Prior to the experiment, students learned AOM, completed an assignment to research and review the existing blockchain methodology and attended a blockchain lecture by the blockchain expert from Tallinn University of Technology, Estonia. This is needed to deliver the knowledge of blockchain to students from the expert.

5.4.3 Result Analysis on Experiment 2

From the scoring system, 45 students' results are calculated and tabulated in Figure 5.3. Based on the results, approximately half of the students could model and analyze at least three lottery industry problems in the computation independent modelling. Most students needed to model the blockchain element in the platform-independent design and modelling. 22.22% of the students could model the minor blockchain element at the design specification.

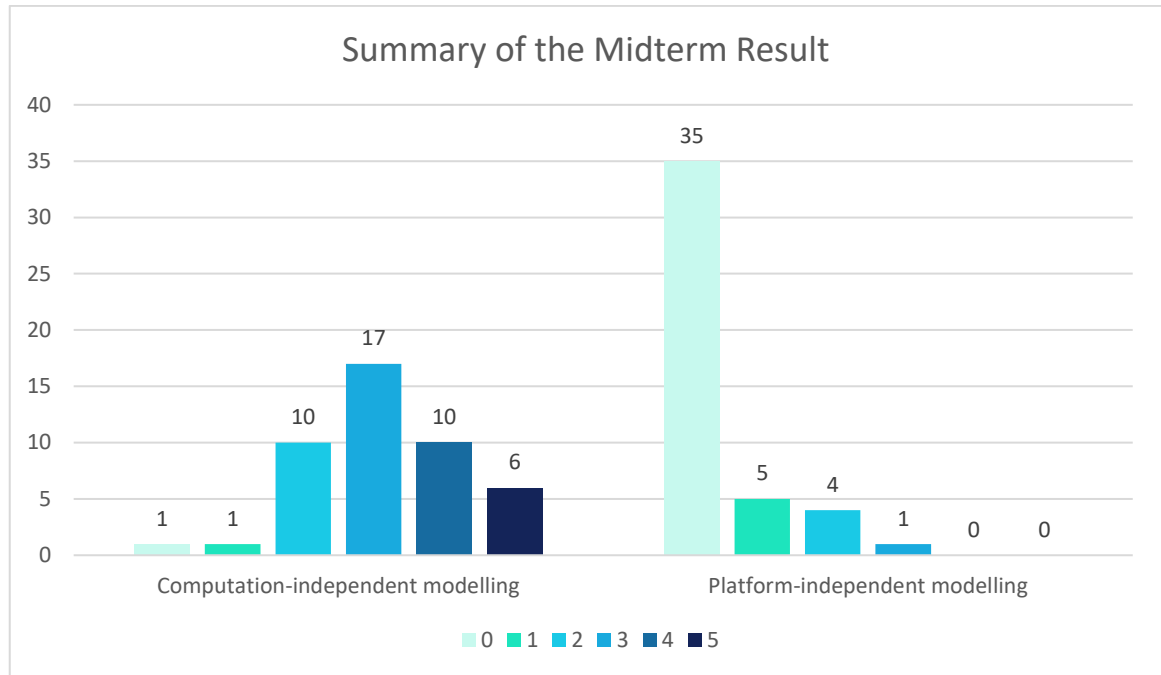


Figure 5.3: The Benefit of AOM to Model Blockchain Application Development

From the findings, we can foresee the gaps when modelling the blockchain application through AOM in detail. The students tend to mix the problem analysis and technology design even though the AOM stated the problem and technology design in two different layers. For example, most of the students included blockchain technology in the computation-independent modelling even though they knew the computation independent modelling was for problem analysis. On the other hand, the students needed to identify the elicited answers based on the case study given. This led to missing blockchain elements during the modelling process.

Although the students learned AOM and obtained blockchain knowledge through lectures and assignments, most students still needed help understanding the different abstractions of AOM. It is also apparent that blockchain properties must be defined in the AOM. Nevertheless, the AOM is too general, making it difficult to trigger the blockchain requirement without the extension of AOM.

To further analyse the usage of AOM, a usability study was conducted among the same students. Based on Figure 5.4, the possible reason for not using the goal model is that it is not useful for a programmer, leading to 24.44% votes. Then followed by it is not useful for clients with 20.00% votes. Next, there were 17.78% votes for not using the goal model because the analysts needed to understand it better, and it captured redundant information. The least possible reasons it could be more useful for programmers garnered 8.89% votes.

28.89% of the students agreed that the role model's ability to capture redundant information is the top reason for not using it. Then 20.00% of the students agreed that role model is not suitable for programmers, 11.11% of the students agreed it is not useful for clients, 8.89% of students agreed it is not useful for most projects, and only 4.44% of students felt that analysts would not well understand it.

The domain model has reported that not being useful with clients is the possible reason for not using it, accumulating 37.78% of student votes. Then is followed by 28.89% of votes stating that the domain model would capture redundant information, 13.33% for not well understood by the analysts, 6.67% for not useful for most projects, and 6.67% votes for not useful for programmers.

37.78% of students declared that the interaction model is not useful with clients, leading to the top reason for not using it. 24.44% of votes follow this for information redundant captured by the interaction model, and 11.11% of votes from the students are not useful for programmers. Lastly, not well understood by analysts and is not useful for most projects got the equal possible reason for not using the interaction model at 6.67%.

The behaviour model has been reported as the possible reason for not using it because it is not useful for clients who accumulated 44.44% of student votes. The following reason

is capturing redundant information by the behaviour model with 20% votes. Then is followed by not useful for programmers and not useful for most projects have equal votes from the students, which are 8.89% votes. Lastly, the least possible for not using it is not well understood by the analysts, with only 6.67% votes.

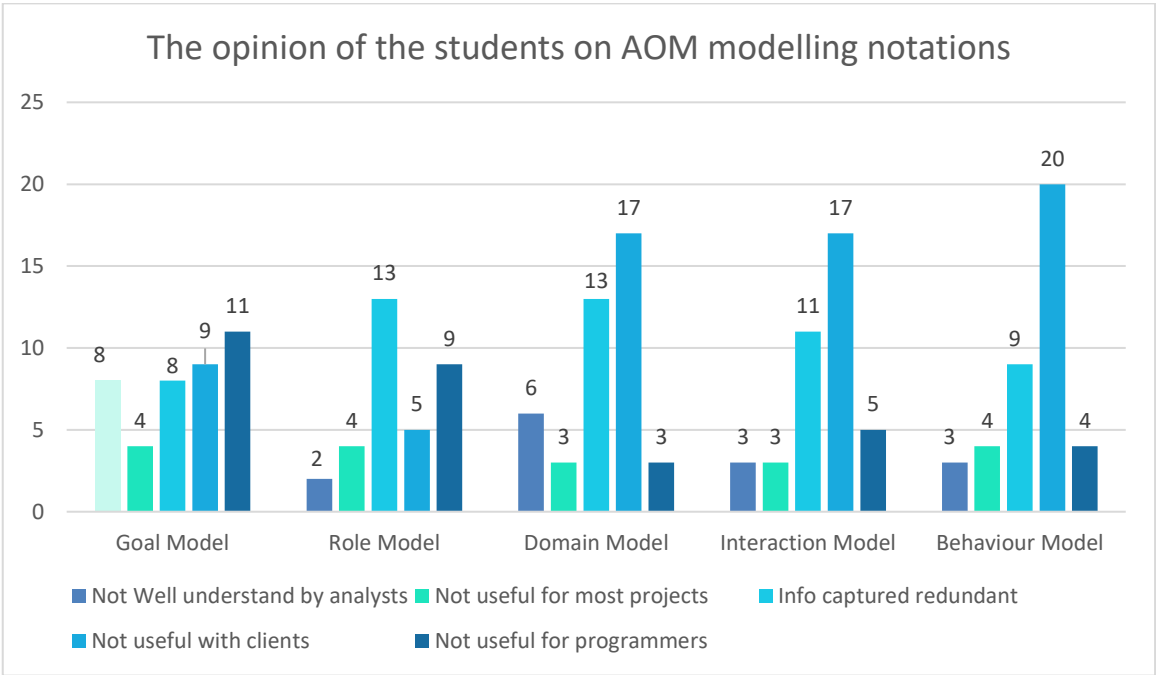


Figure 5.4: The Survey of AOM Modelling Notations in General

The findings show that the generalisation of AOM has reduced the understanding of AOM in blockchain application development. The generalisation behaviour could lead to missing blockchain potential that could help solve problems such as fairness. This is entirely true in adopting AOM to model new technology applications.

As blockchain application is a complex development, there is a need to contextualise AOM to reduce the adoption of blockchain applications among novices. Hence, we introduce an extended AOM, as presented in Chapter 3.

5.5 Experiment 3 – The Usage of Extended AOM for Blockchain Application Development

As mentioned before, AOM is insufficient to model the blockchain application. Hence, an extended AOM was proposed in this study in order to discover and design blockchain-enabling applications. This experiment presents the usability analysis of the extended AOM for blockchain application development.

Overall, there were 45 students involved in Experiment 3. The experiment started with a workshop on extended AOM. The workshop included an introduction to the extended AOM by using the voting system from Chapter 3. Through the workshop, the students understood the extended AOM and learned how to apply the methodology in the blockchain-enabled voting system. After a month, students sat for the final exam in which one of the questions required them to model the case study through extended AOM.

The scoring system for experiment 3 was extended based on experiment 2. Thus, the scoring criteria were extended with phase 3, as shown in Table 5.5.

Table 5.5: Scoring Criteria for Experiment 3

Computation independent modelling		
No	Criteria	Point
1.	Is the model at this layer able to show fairness issues?	1
2.	Is the model at this layer able to show transparency issues?	1
3.	Is the model at this layer able to show scalability issues?	1
4.	Is the model at this layer able to show how bankers manage the issue?	1
5.	Is the model at this layer able to show how intermediaries sell a ticket?	1
Early identification of blockchain use case		

Table 5.5 continued

No	Criteria	Point
1.	Is the model at this layer able to eliminate the middleman?	1
2.	Is the model at this layer able to show the transaction history?	1
3.	Is the model at this layer able to show the peer market structure?	1
Platform independent design and modelling		
No	Criteria	Point
1.	Is the model at this layer able to show how smart contracts solve fairness?	1
2.	Is the model at this layer able to show how Blockchain provide transparency?	1
3.	Is the model at this layer able to show the transaction made through a smart contract?	1
4.	Is the model at this layer able to show how smart contracts arrange the player ticket?	1
5.	Is the model at this layer able to show how data is store on the Blockchain?	1

The final results of the students are shown in Figure 5.5. From the findings in Figure 5.5, the extended AOM helps to derive more potential blockchain criteria. The extended AOM provides more complete steps to help the students identify the blockchain requirement in phase 3 of early identification of blockchain use case.

Thus, the blockchain design is much complete in the platform-independent design and modelling. The result at the computation independent modelling phase is similar to experiment 2. As the provided case study is the same, the students analyzed the domain requirement is similar to in experiment 2. In phase 3, most of the students could capture the needs of blockchain, such as eliminating intermediaries, presenting the market structure, and

the value exchange between the agents. Significantly, over half of the students could model at least three blockchain criteria. 44.44% of the students achieved 4 points and above, 15.55% scored 3 points, 28.89% got 2 points, and only 11.11% got 1 point and below.

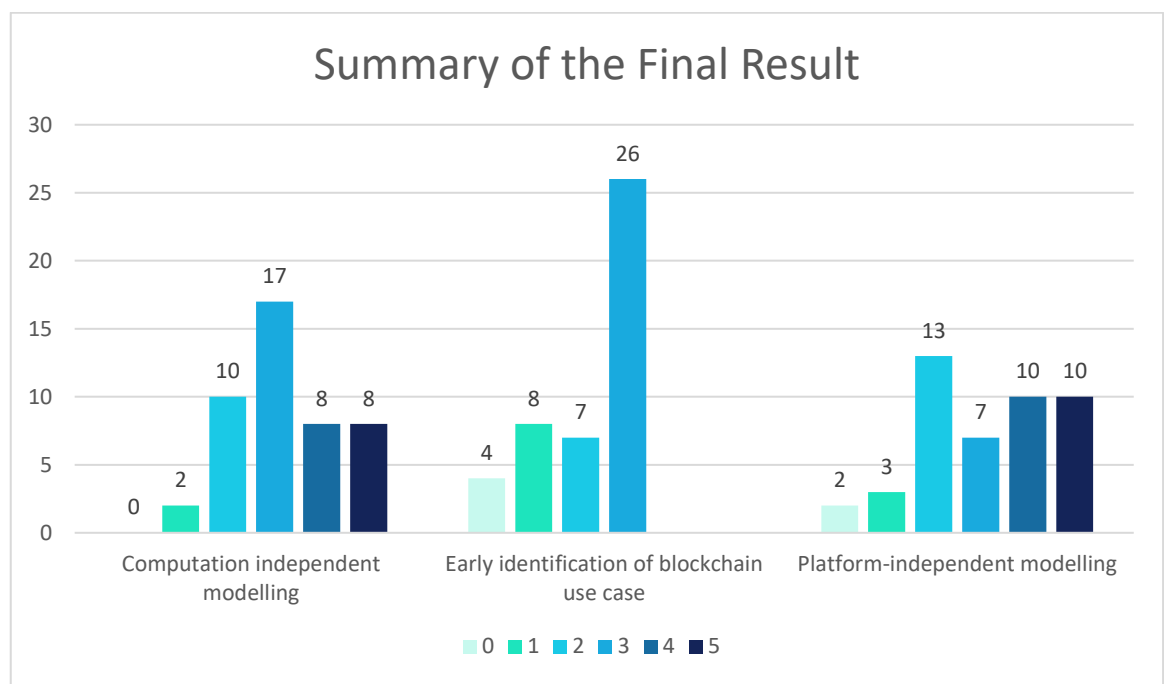


Figure 5.5: The Summary of the Final Result

With the extended AOM, the platform-independent modelling phase results have improved significantly. The students were able to understand how blockchain may handle fairness and transparency. For example, a smart contract could execute the select-winner mechanism to achieve fairness. Some students allowed the smart contract to generate random numbers, and some let the system choose a winner randomly.

The graph in Figure 5.6 shows a clear trend in which the students could completely utilize blockchain technology to solve the problem by extracting more blockchain properties. 60% of the students can model at least three blockchain elements using the extended AOM, whereas only 2.22% of the students achieve this score. The mean score for AOM and extended AOM is tabulated in Table 5.7. Significantly, the mean score of the platform-

independent modelling phase for extended AOM is 2.755, higher than the standard AOM. This finding confirms the usefulness of the extended AOM. This concurs well with the objectives of the research.

Table 5.6: Comparison of Experiment 2 and Experiment 3 in Percentage

	5	4	3	2	1	0
Experiment 2	0.00	0.00	2.22	8.88	11.11	77.78
Experiment 3	22.22	22.22	15.56	28.89	6.66	4.44

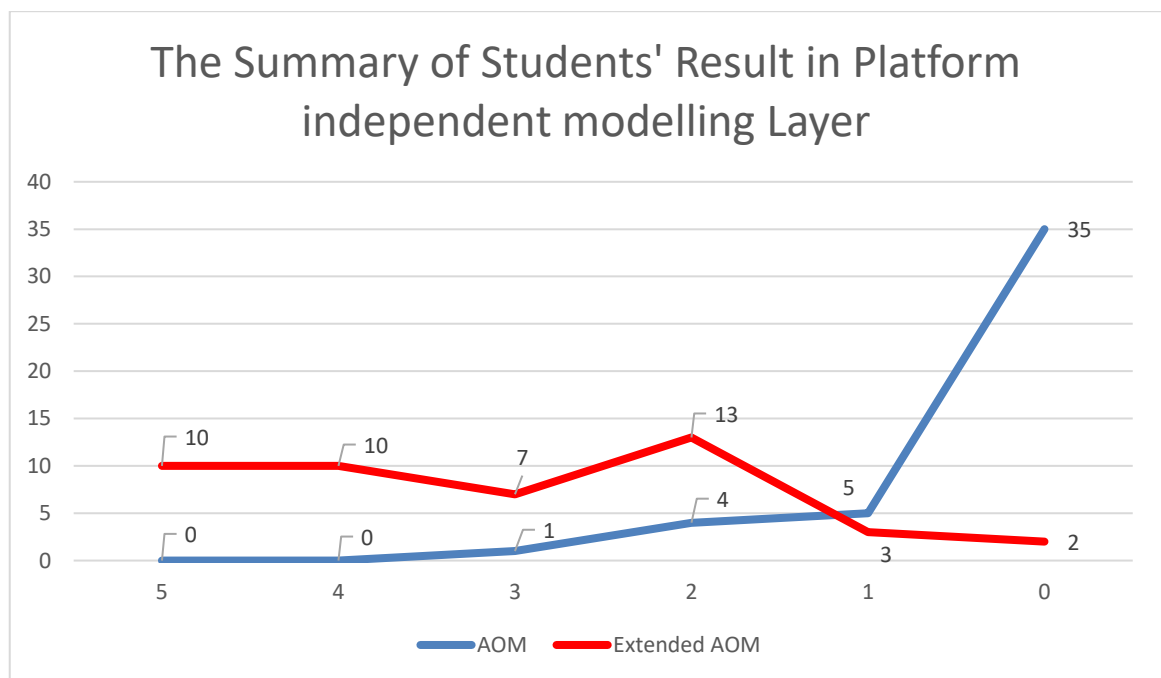


Figure 5.6: The Overview of the Platform Independent Modelling Layer Result

Table 5.7: Mean Score for AOM and Extended AOM

<i>Modelling Technique</i>	<i>AOM</i>	<i>Extended AOM</i>
<i>Mean</i>	0.356	3.111

A survey was conducted to check the usefulness and satisfaction of the extended AOM at the end of Experiment 3. The survey was designed based on the criteria from Seffah, Donyaee, Kline, and Padua (2016). In the survey, the students were required to answer nine questions.

Q1 are to check the adequateness of the experiment time since the case study is part of the test, threatening validity for insufficient time to complete. The question was designed in a dichotomous questions manner. Q2, adopted from Mahunnah (2018), aimed to check the adequateness of the case study in terms of simplicity and understandability by rating between strongly agree to disagree strongly. Q3 to Q9 mainly focused on the extended AOM to investigate the understandability, usability, and resource utilization of the extended AOM based on QUIM model (Quality in Use Integrated Measurement) (Seffah et al., 2016). Understandability, defined as the modelling, is easy to comprehend by the technical and non-technical client. The usability characteristics elaborated by the author are efficiency, effectiveness, satisfaction, usefulness, productivity, and learnability (Seffah et al., 2016).

Table 5.8: The Set of Questions in the Survey

No	Question	Question Type
Q1	I had enough time to complete the exam.	Dichotomous questions
Q2	The blockchain case study is easy to understand.	Rating Questions
Q3	With extended AOM is easier to model blockchain application.	Rating Questions
Q4	The extended AOM is useful for blockchain application development.	Rating Questions
Q5	The goal modelling is used to identify and understand the problem or use case when modelling the Blockchain.	Rating Questions

Table 5.8 continued

Q6	The extended AOM is useful for non-technical clients to understand the blockchain element.	Rating Questions
Q7	The extended AOM is useful for blockchain programming.	Rating Questions
Q8	The importance of the e^3 value model for modelling blockchain.	Checkbox questions
Q9	The strength of the extended AOM.	Checkbox questions

It is interesting to report that most students can understand the case study, as shown in Figure 5.7. The agreement orders ranged from 37.78% strongly agreed that the provided case study is clear and easy to understand, 42.22%, 13.33%, 4.44%, and 2.22%.

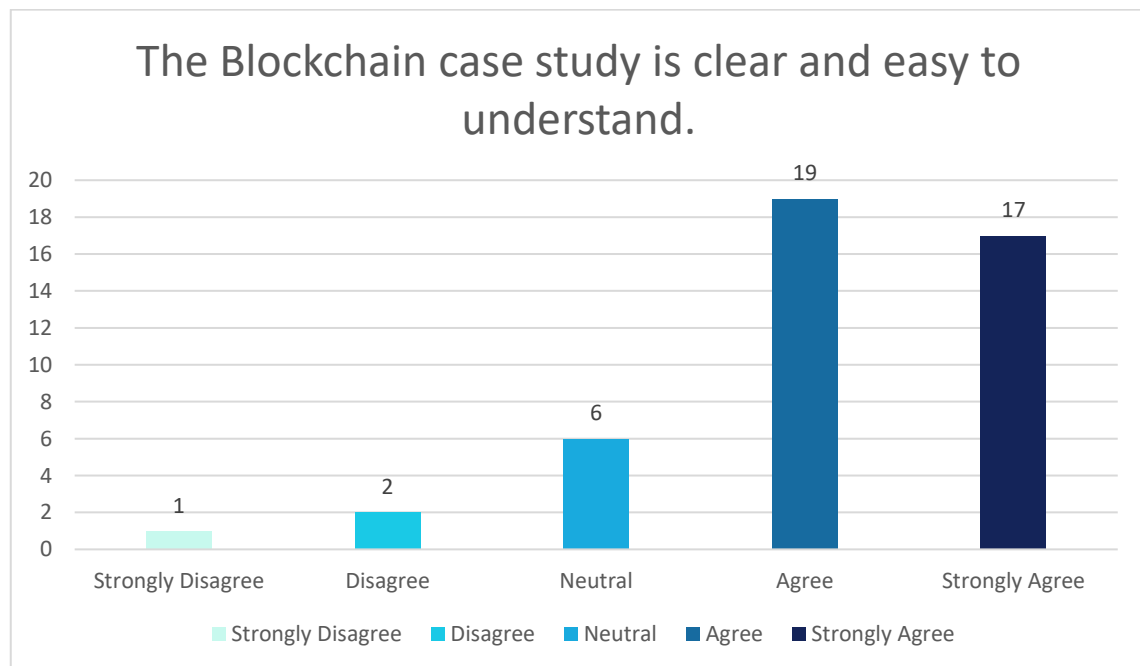


Figure 5.7: The Response of the Understanding of the Case Study

Most students responded positively to extended AOM for blockchain application development as shown in Figure 5.8. With the extended AOM, 20.00% of the students

strongly agreed, 53.33% agreed, 13.33% felt neutral, and 13.33% disagreed with the easiness of modelling blockchain applications with extended AOM.

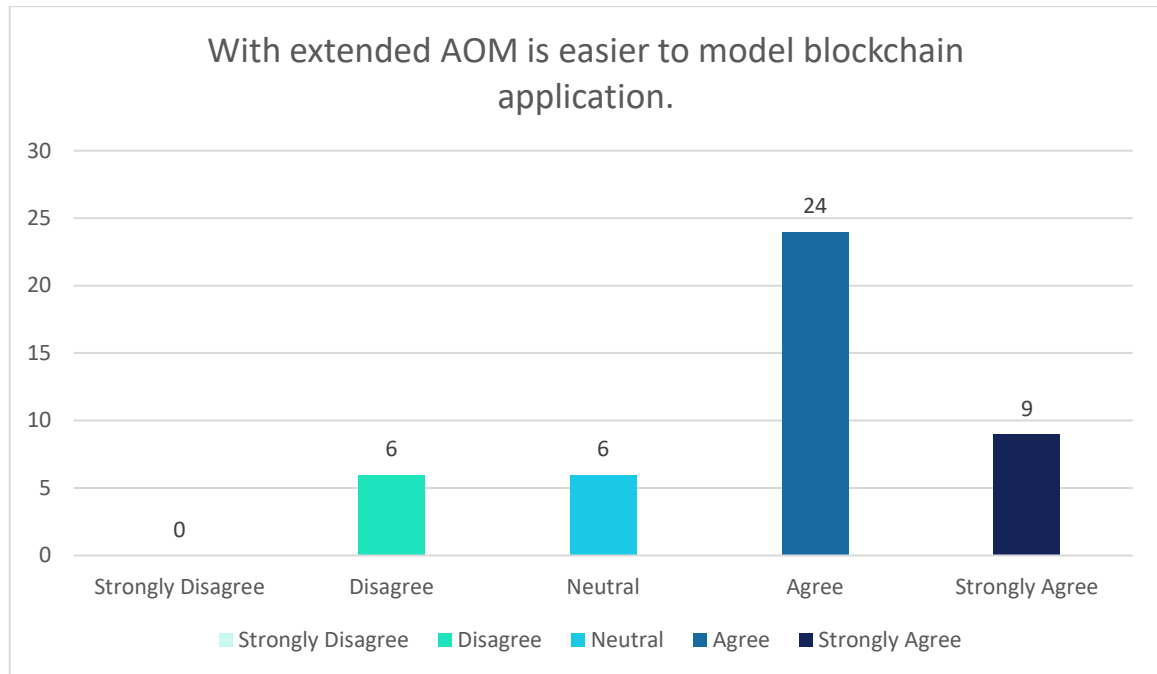


Figure 5.8: The Feedback on the Extended AOM in General

Figure 5.9 shows the usefulness for the development, problem identification, usability for non-technical clients, and implementation of adopting extended AOM in blockchain application development. In response to the usefulness of the application development, 37.78% of students strongly agreed, 48.89% of students agreed, 11.11% of students mentioned neutral, and only 2.22% indicated they disagreed. In the questions on the goal modelling, 46.67% of students commented strongly agree, 35.56% replied agree, 13.33% answered neutral, and only 4.44% disagreed. From the non-technical perspective, most students agree that it is easy to understand and valuable for non-technical clients. The data order from strongly agree to disagree strongly consists of 20.00%, 51.11%, 24.44%, 4.44%, and 0.00%. Lastly, on programming development, over half of the students agreed, which consists of 60.00% of students, 31.11% strongly agreed, 6.66% answered neutral, and only 2.22% disagreed.

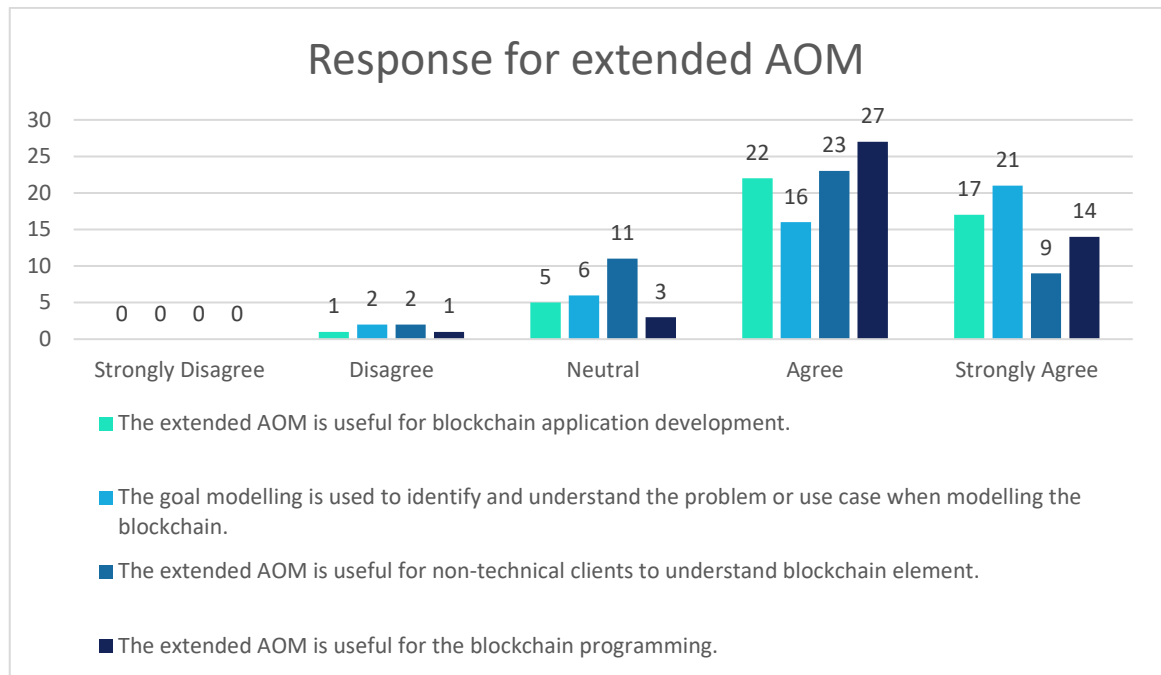


Figure 5.9: The Responses for Extended AOM From Different Perspectives

The usefulness of the e^3 value model was surveyed as shown in Figure 5.10. With the e^3 value model, 84.44% of the participants agreed that the e^3 value model could also help identify more blockchain elements chosen by 75.56% of students. 57.78% of the students agree that the e^3 value can illustrate the peer market structure of the actors, and 46.67% of the students determine that the e^3 value model can present the immutable history by using the value exchange. Lastly, only 2.22% of the student comments that the e^3 value model is hard to understand.

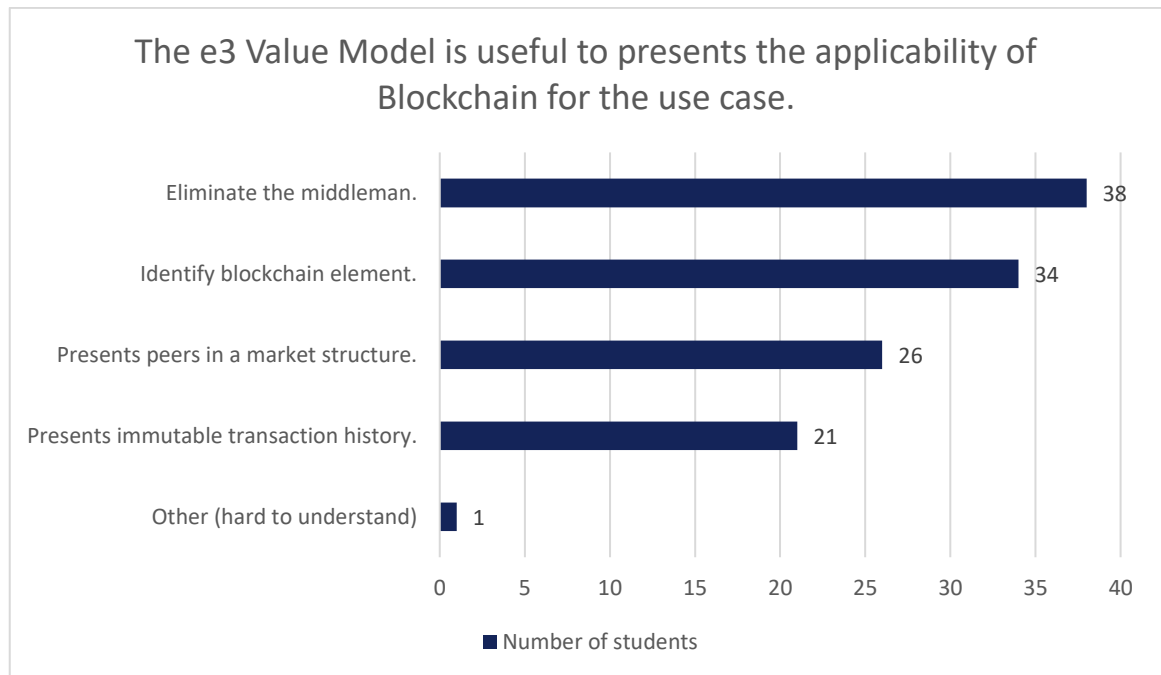


Figure 5.10: The Feedback on the e^3 Value Model

Figure 5.11 shows the strengths of the extended AOM. Most students felt that the extended AOM could help communicate between technical and non-technical clients, consisting of 77.78% of votes. The extended AOM provided the completeness to present the blockchain element and was noted as one of the strengths voted by 75.56% of the students, where 66.67% voted for the correctness of the extended AOM to present the blockchain elements. The extended AOM may enhance the selected blockchain application quality by 57.78% of the students, and 55.56% think it could help enhance productivity. Additionally, the modelling does optimize the code operation, which was rated by 46.67% of students — however, only 2.22% of the students reported that it is not easy to understand.

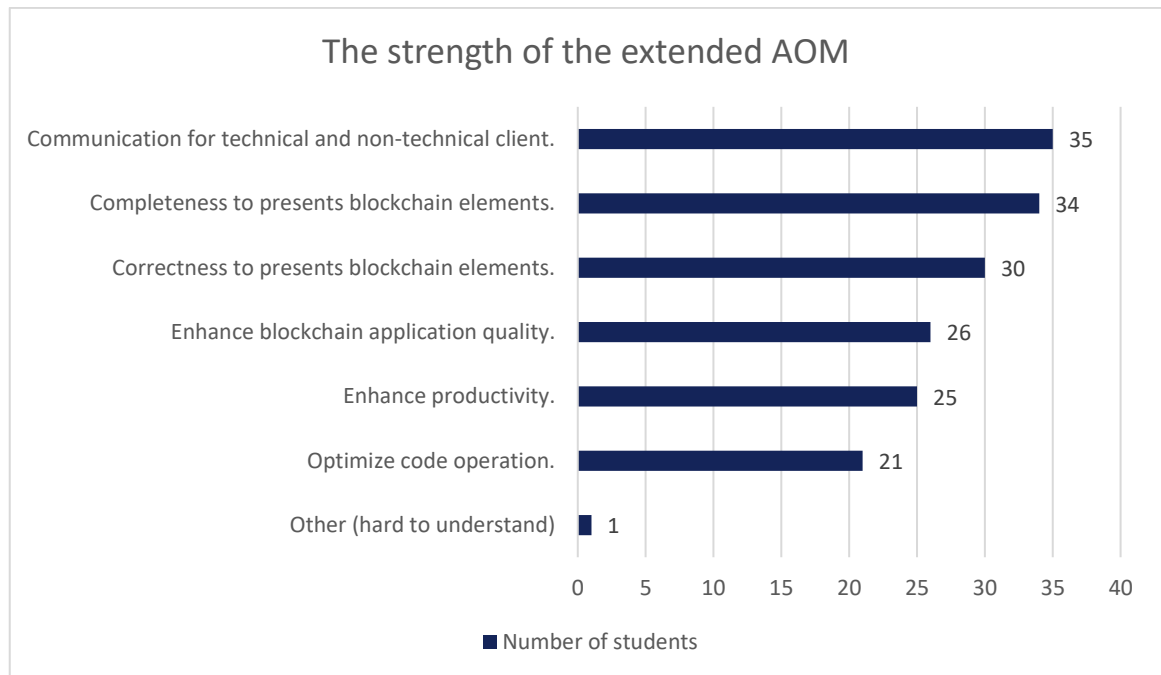


Figure 5.11: The Feedback on the Strength of the Extended AOM

5.6 Discussion

The survey showed a significant positive response to the extended AOM among the students. Each response reflects the usability, effectiveness, and usefulness of the extended AOM. These experiments are in line with the hypothesis stated earlier.

Experiment 1 concludes hypothesis 1, in which the AOM can derive more blockchain requirements than UML. The hypothesis is proved based on Equation 5.1.

Hypothesis 2 was tested, and experiment 2 and experiment 3 results showed that the extended AOM could aid in designing a better blockchain application than AOM. Equation 5.2 is used to prove the hypothesis.

To summarize, the findings in experiment 1 reveal that the AOM is useful for blockchain development. However, the extended AOM could help a beginner design a

blockchain application by starting with problem analysis. The extended AOM outperforms the AOM in investigating and identifying blockchain needs using the e^3 value model.

5.7 Threats to Validity

Several threats to validity should be acknowledged in this study. The students learned blockchain knowledge through lectures. The students that attended the lecture shared a similar grasp of the blockchain concept. They used their understanding of blockchain technology to apply to the UML, AOM and extended AOM. However, the ability of each student to model the case study is different.

Even though students were given blockchain-related assignments and lectures, they may require additional blockchain expertise to reduce threats. To strengthen the blockchain knowledge of the students in experiment 2 and experiment 3, Dr Alexander was invited to give a talk on Blockchain topics. Then the extended AOM was introduced through the workshop to stimulate potential blockchain features to the use case.

Besides, the perspective of each student is different, which may cause threats to validity. The test was given in a blind test mode to reduce the bias of the experiment. Next is the time constraint for the final needed to be increased, and the case study was only part of the exam. Thus, the students might need more time to complete the case study without going into details.

5.8 Conclusion

This chapter discusses the evaluation of the extended AOM through a quantitative analysis. Two experiments were set up on two different batches of novice students. The first experiment aimed to compare the usefulness of UML and AOM. Then the second experiment was intended to specify the AOM issue and proposed the extended AOM to

observe the result. Throughout experiment 2, the tests revealed that the students prefer AOM as the modelling technique for blockchain application development.

Additionally, blockchain knowledge is essential to model the application. To trigger more blockchain properties in the model, we introduced the extended AOM, which leads the students to identify the problem and blockchain properties (remove the middleman, the market structure, and the immutable transaction history) by using the e^3 value model and then apply the properties to the case study. As hypothesized, the experiments proved that the methodology does help the novices to tackle more blockchain potential for a use case.

CHAPTER 6

CONCLUSION

6.1 Overview

This chapter presents the contributions and limitations of the research. The chapter is structured as follows. Section 6.2 presents the contribution of the research; Section 6.3 presents the limitation and future works; and Section 6.4 concludes the thesis.

6.2 Contributions

This research's main contribution is offering new insights into blockchain development through extended AOM. The specific contributions are outlined as follows:

1. The results of UML and AOM are similar. AOM has been found to perform slightly better. Additionally, AOM can serve as an alternative modelling language for blockchain applications, making it a useful tool for developers in this domain.
2. The extended AOM is introduced to develop a blockchain application in a systematic process. The processes include eliciting the problems through HOMER, then mapping these problems to the conceptualisation domain modelling, early identification of blockchain use case, platform-independent design, and platform-specific design.
3. The extended AOM offers design and development support. Compared to the existing methodologies mentioned in Chapter 2 (Agile approach, ABCDE and MDE), the extended AOM provides better expressiveness in terms of blockchain knowledge and role definition, cost-effectiveness, and handling complexity.
4. The extended AOM has been validated using a walkthrough example lottery case study. The process of analysing, designing, and developing blockchain-enabling

lottery games reveals the potential usage of the extended AOM to develop a blockchain application systematically.

5. The extended AOM was evaluated quantitatively with the students. Furthermore, hypothesis testing was used as evidence to claim the usefulness and usability of the extended AOM. Finally, testing the extended AOM for blockchain applications is novel. Previous studies still need extensive discussion centred on the quantitative analysis of AOM among novices for blockchain application development.

6.3 Limitations and Future Work

The constraint of this research that could lead to future works is discussed in this section. The following are some of the limitations that can be explored and considered in this section:

1. This domain study focuses on two simple use cases: voting and win a fortune game. In future, the same study should be applied to a complex system such as blockchain-based supply chain management and blockchain-based electric health record system.
2. The number of blockchain requirements of the study is limited. The research only looks at a few types of blockchain properties. In future, various blockchain requirements should be considered in a more complex system, such as security and consensus.
3. The experiment is restricted to UML, AOM, and extended AOM. The experiments were conducted during the COVID-19 pandemic. various blockchain requirements should be considered in a more complex system, such as security and consensus.
4. The limitation of this methodology is that it is suitable for the Ethereum platform only. In future, we wish to extend the methodology that can serve various blockchain platforms to achieve interoperability. The future methodology will guide the IT teams to select the most appropriate blockchain platform for the solution.

5. The methodology does not include generating test cases from the model. This shall cover the security aspect of the methodology.

6.4 Conclusion

This research has provided new insights into the blockchain development approach. In doing so, we reviewed the current literature review to understand the state of the art on blockchain technology, summarised the existing approach and plotted the gap. Then, extended AOM is introduced for blockchain-based application development. The proposed methodology was validated using a win-a-fortune game case study. Three experiments were conducted with the students to collect evidence to evaluate the extended AOM in terms of comprehensibility, usability, and usefulness. The analysis results revealed that the extended AOM successfully helps the novices to elicitate blockchain requirements along the way from analysing the problems and identifying the blockchain needs.

REFERENCES

- Abbas, M., Rashid, M., Azam, F., Rasheed, Y., Anwar, M. W., & Humdani, M. (2021). A Model-Driven Framework for Security Labs using Blockchain Methodology. *2021 IEEE International Systems Conference (SysCon)*.
<https://doi.org/10.1109/syscon48628.2021.9447125>
- Almeida, S., Albuquerque, A., & Silva, A. (2018). An Approach to Develop Software that Uses Blockchain. *Advances in Intelligent Systems and Computing*, 346–355.
https://doi.org/10.1007/978-3-319-91186-1_36
- Aslam, J., Saleem, A., Khan, N. T., & Kim, Y. B. (2021). Factors influencing blockchain adoption in supply chain management practices: A study based on the oil industry. *Suma de Negocios*, 6(2), 124–134.
- Azogu, I., Norta, A., Papper, I., Longo, J., & Draheim, D. (2019). A framework for the adoption of blockchain technology in healthcare information management systems: A case study of Nigeria. *ACM International Conference Proceeding Series, Part F1481*(March 2020), 310–316.
- Baralla, G., Pinna, A., & Corrias, G. (2019). Ensure Traceability in European Food Supply Chain by Using a Blockchain System. *2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*.
<https://doi.org/10.1109/wetseb.2019.00012>
- Bosu, A., Iqbal, A., Shahriyar, R., & Chakraborty, P. (2019). Understanding the motivations, challenges and needs of Blockchain software developers: a survey. *Empirical Software Engineering*, 24(4), 2636–2673. <https://doi.org/10.1007/s10664-019-09708-7>.

- Bousetta, B., El Beggar, O., & Gadi, T. (2013). A methodology for CIM modelling and its transformation to PIM. *Journal of Information Engineering and Applications*, 3(2), 1–22.
- Bumblauskas, D., Mann, A., Dugan, B., & Rittmer, J. (2020). A blockchain use case in food distribution: Do you know where your food has been? *International Journal of Information Management*, 52, 102008. <https://doi.org/10.1016/j.ijinfomgt.2019.09.004>
- Chakraborty, P., Shahriyar, R., Iqbal, A., & Bosu, A. (2018). Understanding the software development practices of blockchain projects: A survey. *International Symposium on Empirical Software Engineering and Measurement*.
- Cheah, W. S. (2010). Patterns for Agent-oriented Software Development (Doctoral dissertation, University of Melbourne, Department of Computer Science and Software Engineering).
- Choi, T. M., & Ouyang, X. (2021). Initial coin offerings for blockchain based product provenance authentication platforms. *International Journal of Production Economics*, 233(November 2020), 107995.
- Cocco, L., Mannaro, K., Tonelli, R., Mariani, L., Lodi, M. B., Melis, A., . . . Fanti, A. (2021). A Blockchain-Based Traceability System in Agri-Food SME: Case Study of a Traditional Bakery. *IEEE Access*, 9, 62899–62915. <https://doi.org/10.1109/access.2021.3074874>
- Crawford, S. E. S., & Ostrom, E. (1995). A Grammar of Institutions. *American Political Science Review*, 89(3), 582–600. <https://doi.org/10.2307/2082975>
- Daud, N., Abdinasir, A., Moallim, A., Sham, N., & Abu, A. (2022). Sustainable Social Bank Development using Blockchain. *International Journal on Perceptive and Cognitive*

Computing (IJPCC), 8(1), 53–58.

- Deshpande, A., Stewart, K., Lepetit, L., & Gunashekar, S. (2017). Understanding the landscape of Distributed Ledger Technologies/Blockchain: Challenges, opportunities, and the prospects for standards. *Understanding the Landscape of Distributed Ledger Technologies/Blockchain: Challenges, Opportunities, and the Prospects for Standards*.
- 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.
- Fahmideh, M., Grundy, J., Ahmad, A., Shen, J., Yan, J., Mougouei, D., . . . Abedin, B. (2022). Engineering Blockchain-based Software Systems: Foundations, Survey, and Future Directions. *ACM Computing Surveys*, 55(6), 1–44.
<https://doi.org/10.1145/3530813>
- Farnoush, A., Gupta, A., Dolarsara, H. A., Paradice, D., & Rao, S. (2021). Going beyond intent to adopt Blockchain: an analytics approach to understand board member and financial health characteristics. *Annals of Operations Research*, 308(1–2), 93–123.
<https://doi.org/10.1007/s10479-021-04113-0>
- Ferre, X. (2010). *Handbook of Software Engineering and Knowledge Engineering - Vol 1: Fundamentals. Handbook of Software Engineering and Knowledge Engineering - Vol 1: Fundamentals*.
- Frantz, C. K., & Nowostawski, M. (2016). From Institutions to Code: Towards Automated Generation of Smart Contracts. *2016 IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W)*. <https://doi.org/10.1109/fas-w.2016.53>

- Fridgen, G., Lockl, J., Radszuwill, S., Rieger, A., Schweizer, A., & Urbach, N. (2018). A solution in search of a problem: A method for the development of blockchain use cases. *Americas Conference on Information Systems 2018: Digital Disruption, AMCIS 2018*, 4801.
- Gordijn, J., Wieringa, R. J., Ionita, D., & Kaya, F. (2019). Towards a sustainable blockchain use case. *Proceedings VMBO Workshop 2019*, (March), 1–5.
- Haeri, S. H., Thompson, P., Davies, N., Van Roy, P., Hammond, K., & Chapman, J. (2022). Mind Your Outcomes: The Δ QSD Paradigm for Quality-Centric Systems Development and Its Application to a Blockchain Case Study. *Computers*, 11(3), 45.
- Hametner, R., Winkler, D., Östreicher, T., Biffl, S., & Zoitl, A. (2010). The adaptation of test-driven software processes to industrial automation engineering. *IEEE International Conference on Industrial Informatics (INDIN)*, (May 2014), 921–927.
- Huemer, C., Schmidt, A., Werthner, H., & Zapletal, M. (2008). A UML profile for the e3-Value e-business model ontology. *CEUR Workshop Proceedings*, 336(February), 1–15.
- Ibba, S., Pinna, A., Lunesu, M. I., Marchesi, M., & Tonelli, R. (2018). Initial coin offerings and agile practices. *Future Internet*, 10(11), 1–21.
- Ibba, S., Pinna, A., Seu, M., & Pani, F. E. (2017). CitySense. *Proceedings of the XP2017 Scientific Workshops*. <https://doi.org/10.1145/3120459.3120472>
- Jairam, S., Gordijn, J., Da Silva Torres, I., Kaya, F., & Makkes, M. (2021). A decentralized fair governance model for permissionless blockchain systems. *CEUR Workshop Proceedings*, 2835, 23–31.
- Jia, Z., Chen, R., & Li, J. (2019). DeLottery. *Proceedings of the 2019 2nd International*

Conference on Blockchain Technology and Applications.

<https://doi.org/10.1145/3376044.3376049>

Jiang, S., Jakobsen, K., Bueie, J., Li, J., & Haro, P. H. (2022). A Tertiary Review on Blockchain and Sustainability With Focus on Sustainable Development Goals. *IEEE Access*, 10, 114975–115006. <https://doi.org/10.1109/access.2022.3217683>

Jurgelaitis, M., Butkienė, R., Vaičiukynas, E., Drungilas, V., & Čeponienė, L. (2019). Modelling principles for blockchain-based implementation of business or scientific processes. *CEUR Workshop Proceedings*, 2470, 43–47.

Kardoš, M., & Drozdová, M. (2010). Analytical method of CIM to PIM transformation in model driven architecture (MDA). *Journal of Information and Organizational Sciences*, 34(1), 89–99.

Kim, H. M., & Laskowski, M. (2018). Toward an ontology-driven blockchain design for supply-chain provenance. *Intelligent Systems in Accounting, Finance and Management*, 25(1), 18–27.

Knezevic, D. (2018). Impact of blockchain technology platform in changing the financial sector and other industries. *Montenegrin Journal of Economics*, 14(1), 109–120.

Kombe, C., Ally, M., & Sam, A. (2018). A review on healthcare information systems and consensus protocols in blockchain technology. *International Journal of Advanced Technology and Engineering Exploration*, 5(49), 473–483.

Lallai, G., Pinna, A., Marchesi, M., & Tonelli, R. (2020). Software Engineering for DApp Smart Contracts Managing Workers Contracts. *DLT@ITASEC*.

Lenarduzzi, V., Lunesu, M. I., Marchesi, M., & Tonelli, R. (2018). Blockchain applications for Agile methodologies. *ACM International Conference Proceeding Series, Part*

F1477(May).

- Li, G., & Fang, C.-C. (2021). Promoting Information-Resource Sharing within the Enterprise: A Perspective of Blockchain Consensus Perception. *Journal of Open Innovation: Technology, Market, and Complexity*, 7(3), 177.
- Liao, D. Y., & Wang, X. (2017). Design of a Blockchain-Based Lottery System for Smart Cities Applications. *Proceedings - 2017 IEEE 3rd International Conference on Collaboration and Internet Computing, CIC 2017, 2017-Janua*, 275–282.
- Lin, I. C., & Liao, T. C. (2017). A survey of blockchain security issues and challenges. *International Journal of Network Security*, 19(5), 653–659.
- Lund, E. H., Jaccheri, L., Li, J., Cico, O., & Bai, X. (2019). Blockchain and sustainability: A systematic mapping study. *Proceedings - 2019 IEEE/ACM 2nd International Workshop on Emerging Trends in Software Engineering for Blockchain, WETSEB 2019*, 16–23.
- Mahunnah, M. (2018). An Empirical Evaluation of the Requirements Engineering Tool for Socio-Technical Systems. *2018 IEEE 7th International Workshop on Empirical Requirements Engineering (EmpiRE)*, 8–15.
- Mannaro, K., Pinna, A., & Marchesi, M. (2017). Crypto-Trading: Blockchain-oriented energy market. *2017 AEIT International Annual Conference: Infrastructures for Energy and ICT: Opportunities for Fostering Innovation, AEIT 2017, 2017-Janua*, 1–5.
- Marchesi, M., Marchesi, L., & Tonelli, R. (2018). An Agile Software Engineering Method to Design Blockchain Applications. *Proceedings of the 14th Central and Eastern European Software Engineering Conference Russia*.

<https://doi.org/10.1145/3290621.3290627>

- Marchesi, L., Marchesi, M., & Tonelli, R. (2020). ABCDE—agile block chain DApp engineering. *Blockchain: Research and Applications*, 1(1–2), 100002. <https://doi.org/10.1016/j.bcra.2020.100002>
- Miller, T., Lu, B., Sterling, L., Beydoun, G., & Taveter, K. (2014). Requirements elicitation and specification using the agent paradigm: The case study of an aircraft turnaround simulator. *IEEE Transactions on Software Engineering*, 40(10), 1007–1024.
- Miraz, M.H., & Ali, M. (2020). Blockchain Enabled Smart Contract Based Applications: Deficiencies with the Software Development Life Cycle Models. *Information Systems eJournal*.
- Montes, J. M., Larios, V. M., Avalos, M., & Ramírez, C. E. (2018). Applying Blockchain to Supply Chain Operations at IBM Implementing Agile Practices in a Smart City Environment. *Research in Computing Science*, 147(2), 65–75.
- Norta, A., & Udokwu, C. (2019). A Blockchain System for the Attestation and Authorization of Digital Assets A Decentralized Blockchain System for the Attestation and Authorization of Digital Assets, (July).
- Nzuva, S. (2019). Smart Contracts Implementation, Applications, Benefits, and Limitations. *Public Policy and Administration Research*, 63–75.
- Paatey, E., & Ofori-Dwumfuo, G. O. (2011). The design of an electronic voting system. *Research Journal of Information Technology*, 3(2), 91–98.
- Poels, G., Kaya, F., Verdonck, M., & Gordijn, J. (2019). Early identification of potential distributed ledger technology business cases using e3value Models. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and*

Lecture Notes in Bioinformatics), 11787 LNCS(October 2017), 70–80.

Porru, S., Pinna, A., Marchesi, M., & Tonelli, R. (2017). Blockchain-oriented Software Engineering: Challenges and New Directions. *Ieee/Acm*, 1–4.

Rocha, H., & Ducasse, S. (2018). Preliminary steps towards modeling blockchain oriented software. *Proceedings - International Conference on Software Engineering*, 52–57.

Saichua, P., Khunthi, S., & Chomsiri, T. (2019). Design of blockchain lottery for Thai government. *ECTI DAMT-NCON 2019 - 4th International Conference on Digital Arts, Media and Technology and 2nd ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering*, (January), 9–12.

Seffah, A., Donyaee, M., Kline, R. B., & Padda, H. K. (2006). Usability measurement and metrics: A consolidated model. *Software Quality Journal*, 14(2), 159–178.
<https://doi.org/10.1007/s11219-006-7600-8>

Sterling, L. S., & Taveter, K. (2009). *The Art of Agent-Oriented Modeling*.
<https://doi.org/10.7551/mitpress/7682.001.0001>

Sukheja, D., Indira, L., Sharma, P., & Chirgaiya, S. (2019). Blockchain technology: A comprehensive survey. *Journal of Advanced Research in Dynamical and Control Systems*, 11(9 Special Issue), 1187–1203.

Udokwu, C., Anyanka, H., & Norta, A. (2020). Evaluation of Approaches for Designing and Developing Decentralized Applications on Blockchain. *Proceedings of the 2020 4th International Conference on Algorithms, Computing and Systems*.
<https://doi.org/10.1145/3423390.3426724>

Waller, V., Johnston, R. B., & Milton, S. K. (2008). A methodology for situated analysis and design. *Handbook of Research on Modern Systems Analysis and Design*

Technologies and Applications, 23–34.

- Wilmann, D., & Sterling, L. (n.d.). Guiding agent-oriented requirements elicitation: HOMER. *Fifth International Conference on Quality Software (QSIC'05)*.
<https://doi.org/10.1109/qsic.2005.34>
- Xu, X., Weber, I., & Staples, M. (2019). Model-Driven Engineering for Blockchain Applications. *Architecture for Blockchain Applications*, 149–172.
https://doi.org/10.1007/978-3-030-03035-3_8
- Zambonelli, F., Jennings, N. R., & Wooldridge, M. (2001). Organisational Abstractions for the Analysis and Design of Multi-agent Systems. *Agent-Oriented Software Engineering*, 235–251. https://doi.org/10.1007/3-540-44564-1_16
- Zeng, Y., & Zhang, Y. (2019). Review of research on blockchain application development method. *Journal of Physics: Conference Series*, 1187(5).

APPENDICES

Appendix A: Journal Publications

1. Ten, M. L. B., Cheah, W. S., Khairuddin, M. A., Mit, E., & Erianda, A. (2021). Agent-oriented modelling for Blockchain Application Development: Feasibility Study. *JOIV : International Journal on Informatics Visualization*, 5(3), 248. <https://doi.org/10.30630/joiv.5.3.670>
2. Ten, M. L. B., Cheah, W. S., & Sim, Y. W. (2022). Engineering blockchain enabling win a fortune game among novice through eAOM. *14th ACM Web Science Conference 2022*. <https://doi.org/10.1145/3501247.3539018>

Appendix B: Blockchain Quiz and Answer During the Workshop

No	Question	Options	Answer
1	What is a blockchain?	A distributed ledger on a peer-to-peer network	√
		An exchange	
		A type of cryptocurrency	
		A centralized ledger	
2	What is the ledger type used by Bitcoin?	Decentralised ledger	
		Distributed ledger	
		The distributed and decentralised ledger	√
		None of the above	
		Yes– but only within a certain time	
		No – they cannot be altered.	√

Appendix A continued

3	What does a block in a blockchain consist of?	The hash point, timestamp, transaction data	√
		The hash point, IP of the owner, transaction data	
		Blockchain name, IP of the owner, transaction data	
		The hash point, timestamp, IP of the owner	
4	Once records are submitted on a blockchain, can they be altered?	Yes – the user can go back in and alter them at any time.	
5	What is immutability?	Running without the concept of human trust.	
		Data in a block in a blockchain cannot be tampered with	√
		Peer to peer digital ledger of transaction	
		The majority of nodes need to come to a consensus.	
6	What does DApp mean?	Downloadable Application	
		Distributed Application	
		Decentralised Application	√
		Dintermediate Application	
7	What makes a good Blockchain use case?	An identifiable business network	
		A business problem to be solved	
		The need for trust	
		All of these	√

Appendix A continued

8	A smart contract is ...	Business rules are implied by the contract embedded in Blockchain.	√
		A legal contract that is written in constrained English	
		A cognitive contract	
		None of these	
9	Which of the following is true?	A smart contract is a computer code	
		A smart contract allows the exchange of any asset (money and real estate)	
		A smart contract keeps the anonymity of the involved parties.	
		All of the above	√
10	Who takes the role of a mediator party when using a smart contract?	Virtual lawyers	
		Intermediate banks	
		Miners	
		There is no mediator	√
11	What is used to accessing a wallet?	Private key	√
		Public Key	
		Seed	
		Password	
		The name of a peer-to-peer network	
		A commercial blockchain-based protocol featuring a smart contract manager	
		A blockchain distributed platform featuring smart contract functionality	√

Appendix A continued

12	A blockchain distributed platform featuring smart contract functionality of...	The name of a public Blockchain	
13	What is the meaning of a private key?	A key to open a secret door	
		A key not given to the public	√
		A key on your keychain	
		The master key	
14	What does Asymmetric Encryption use?	The public and private key	√
		Proof of stake	
		Public key only	
		Private key only	
15	Cryptocurrency is store on?	Compact Disk	
		Wallet	√
		Bank Account	
		USB	
16	What is the gas used for?	Ethereum	
		Bitcoin	
		Ripple	
		Transaction fee	√