Blockchain-based Decentralized Mechanism for Conversation system

by

Wenli Yang, PhD

School of Information and Communication Technology

College of Sciences and Engineering

Submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

Supervisors:

Prof. Byeong Kang

Dr. Saurabh Garg

Dr. Winyu Chinthammit

University of Tasmania

March 2022

Declaration of Originality

I declare that this thesis contains no material which has been accepted for a degree or diploma by the University or any other institution, except by way of background information and duly acknowledged in the thesis, and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due acknowledgement is made in the text of the thesis.

Signed: ___Wenli Yang, PhD___

Date:17/03/2022

Statement of Authority of Access

This thesis may be made available for loan and limited copying in accordance with the *Copyright Act* 1968.

Statement regarding published work contained in thesis

The publishers of the papers comprising Chapters 2,3,4,5 and 6 hold the copyright for that content and access to the material should be sought from the respective journals/publishers. Due to the inclusion of published material there is unavoidable repetition of material between chapters in this thesis. The remaining nonpublished content of the thesis may be made available for loan and limited copying and communication in accordance with the Statement of Access and the *Copyright Act* 1968.

Signed: Wenli Yang, PhD

Date: 17/03/2022

List of Publications in this Research

Wenli Yang, Saurabh Garg, Ali Raza, David Herbert, Byeong Kang, "Blockchain: trends and future", Proceedings of the 15th Pacific Rim Knowledge Acquisition Workshop: Knowledge Management and Acquisition for Intelligent Systems, 2018, pp: 201-210.

Wenli Yang, Erfan Aghasian, Saurabh Garg, David Herbert, Leandro Disiuta, Byeong Kang, " A survey on blockchain-based internet service architecture: Requirements, challenges, trends, and future", IEEE Access 7, 2019, 75845-75872.

Wenli Yang, Saurabh Garg, Zhiqiang Huang, Byeong Kang, "A decision model for blockchain applicability into Knowledge-Based conversation system ", Knowledge-Based Systems, 2020, 220: 106791.

Wenli Yang, Saurabh Garg, Zhiqiang Huang, Byeong Kang, " Hybrid consensus algorithm for master–slave blockchain in multi-domain conversation system", Expert systems with Applications (under review).

Wenli Yang, Saurabh Garg, Bai Quan, Byeong Kang, "Blockchain-based Decentralized Mechanism for Knowledge Fusion in multi-expert Conversation system.", Expert systems with Applications (under review).

Anything that can conceive of as a supply chain, blockchain can vastly improve its efficiency-it doesn't matter if its people, numbers, data, money.

— Ginni Rometty, CEO IBM

Chester Buckenmaier III, MD, COL (ret), MC, USA, 2021

The future depends on some graduate student who is deeply suspicious of everything I have said."

— Geoffrey Hinton, Godfather of "AI"

University of Toronto

ABSTRACT

Conversation systems are intelligent agents that can help users finish tasks more efficiently via text or spoken interactions, which are among the core technologies in the field of artificial intelligence. The existing conversation systems always need humans or machine learning to maintain the conversation scenarios. To ensure the quality of conversation contents, it is better to use multiple experts group to share and maintain the conversation contents together.

In the context of group work, it is important to build fair and trustworthy incentives as rewards. The existing incentive schemes for conversation systems are decided by administrators or managers. Such schemes rely on certain authorities rather than on a consensus for all participants, and the rewards may not be fair or trustworthy. In addition, the contributions are critical for calculating the incentives of all participants, and the conversation contents used to assign the contributions should be protected. In the traditional conversation system, such information is always stored in log files based on a centralized server, which can provide audit trail, but can be easily erased or alterable without trace, and the centralized server also have high privacy risks, providing attackers a single target to hack. Due to these reasons, existing approaches face several issues such as unfair incentive schemes, contributions tampering as well as privacy problems. These inherent fundamental issues in current conversation systems are concerning topics.

Blockchain has shown its potential of solving these issues with its key features: autonomous and decentralized processing, smart contractual enforcement of goals, traceable trustworthiness in tamper proof transactions, etc. With the development of the blockchain technology, the value of blockchain lies not only to hold crypto-currencies, but allow in integrating significant panoply information over the same platform in a decentralized and secure way. Although existing projects have opened many doors, but the integration of the conversation system with a blockchain is still in an early prototype stage, and many essential characteristics, such as blockchain interoperability, consensus protocol, and incentive schemes need to be designed and integrated to secure conversation management. Herein, we aim to present a novel blockchain-based decentralized conversation system, that can provide trustworthy and effective

conversational services. The key research findings and contributions of this study are:

1) A comprehensive state-of-the-art survey was conducted on the current situations of how the blockchain technology to secure a conversation system and the vision of building a blockchain-based architecture and key technical requirements for building a decentralized conversation system to aid further designing and implementation.

2) Analyzed and identified the requirements of conversation systems and presented a decision model for identifying the best fitting blockchain platform for the conversation systems, which is the foundation of the following designing and development.

3) A novel master–slave chain model for the conversation system was designed and applied to process multiple conversation interactions concurrently from different domains.

4) A new hybrid consensus algorithm for our master–slave chain was proposed herein to achieve collaborative sharing and maintenance validation, and an incentive scheme was designed to generate both economic and non-economic rewards for all nodes participating in the proposed consensus process.

5) A decentralized knowledge-fusion scheme with blockchain-enabled smart contracts was implemented based on the proposed blockchain structure, consensus protocol, and incentive scheme. Furthermore, multiple case studies were utilized to evaluate the feasibility and effectiveness of the blockchain-based decentralized conversation system.

Statement of Co-Authorship

The following people and institutions contributed to the publication of work undertaken as part of this thesis:

Candidate – Wenli Yang,School of Information and Communication Technology

Author 1 – Byeong Kang,Univeristy of Tasmania

Author 2 – Saurabh Garg, Univeristy of Tasmania

Author 3 – Ali Raza, Univeristy of Tasmania

Author 4 – David Herbert, Univeristy of Tasmania

Author 5 – Erfan Aghasian, Univeristy of Tasmania

Author 6 – Leandro Disiuta, Univeristy of Tasmania

Author 7 – Zhiqiang Huang, Univeristy of Tasmania

Author 8 – Quan Bai, Univeristy of Tasmania

Contribution of work by co-authors for each paper:

PAPER 1: Located in Chapter 2

Wenli Yang, Saurabh Garg, Ali Raza, David Herbert, Byeong Kang, "Blockchain: trends and future", Proceedings of the 15th Pacific Rim Knowledge Acquisition Workshop: Knowledge Management and Acquisition for Intelligent Systems, pp: 201-210. Chapter 2 is derived from this publication

Author contributions:

Planning, conducting, and reporting the research: Candidate, Author 1, Author 2

Analysed the data: Candidate

Wrote the manuscript: Candidate, Author 1, Author 2

Critical revision of the paper: Candidate, Author 3, Author 4

Refinement of the paper: Candidate, Author 1, Author 4

PAPER 2: Located in Chapter 2

Wenli Yang, Erfan Aghasian, Saurabh Garg, David Herbert, Leandro Disiuta, Byeong Kang, " A survey on blockchain-based internet service architecture: Requirements, challenges, trends, and future", IEEE Access 7, 75845-75872

Author contributions:

Conceived and designed the review: Candidate, Author 1, Author 2
Analysed the data: Candidate, Author 2
Wrote the manuscript: Candidate, Author 5
Critical revision of the paper: Candidate, Author 5, Author 6
Refinement of the paper: Candidate, Author 2, Author 4

PAPER 3: Located in Chapter 3

Wenli Yang, Saurabh Garg, Zhiqiang Huang, Byeong Kang, "A decision model for blockchain applicability into Knowledge-Based conversation system ", Knowledge-Based Systems, 220, 106791

Author contributions:

Designing, implementation and conducting experiments: Candidate, Author 2
Analysed the data: Candidate, Author 7
Wrote the manuscript: Candidate
Critical revision of the paper: Candidate, Author 5, Author 6
Refinement of the paper: Candidate, Author 1, Author 2

PAPER 4: Located in Chapter 4 and Chapter 5

Wenli Yang, Saurabh Garg, Zhiqiang Huang, Byeong Kang, " Hybrid consensus algorithm for master–slave blockchain in multi-domain conversation system", Expert systems with Applications (under review)

Author contributions:

Designing, implementation and conducting experiments: Candidate, Author 1, Author 2
Analysed the data: Candidate, Author 7
Wrote the manuscript: Candidate

PAPER 5: Located in Chapter 6

Wenli Yang, Saurabh Garg, Bai Quan, Byeong Kang, "Blockchain-based Decentralized Mechanism for Knowledge Fusion in multi-expert Conversation system.", Expert systems with Applications (under review)

Author contributions:

Designing, implementation and conducting experiments: Candidate, Author 1, Author 2, Author 8
Analysed the data: Candidate
Wrote the manuscript: Candidate

We, the undersigned, endorse the above stated contribution of work undertaken for each of the published (or submitted) peer-reviewed manuscripts contributing to this thesis:

Signed:

Wenli Yang	Byeong Kang	Anna Shillabeer
Candidate	Primary Supervisor	Head of School
School of Information and	School of Information and	School of Information and
Communication Technology	Communication Technology	Communication Technology
University of Tasmania	University of Tasmania	University of Tasmania

Date: 17/03/2022

ACKNOWLEDGMENTS

More than three years of doctoral student life have passed in an instant. When looking back on the years passed, I feel a heartfelt sense of fulfillment, and now that my thesis is about to be completed, I feel a lot of emotion. Firstly, I would like to sincerely thank my primary supervisor Professor Kang for his hard work from the beginning to the end of the research topic selection, algorithm design and verification, as well as the ultimate finalisation of the thesis. With his generous and benevolent mind, and positive and optimistic attitude towards research, he has set a lifelong example for me to learn from; indeed, his teachings and encouragement will inspire me to make great efforts and innovate on the road of research and education.

In addition, I would like to thank my two other associate supervisors from my sincerest heart; I thank them for their full support of my research and thesis, and for providing me with all the requisite conditions for research. It can be categorically said that without them, there would be none of today's outcomes. Thank you to Dr. Saurabh Garg specifically for providing me with kind and patient guidance every time I have encountered difficulties in writing my papers – the advice on all things has been invaluable.

Thank you too to Dr. Winyu Chinthammit for helping me with solving ideas and also aiding me with the proofreading of both the introduction and conclusion of my thesis. Furthermore, I also wish to express my thanks to Professor Quan Bai for discussing with me my algorithm experiment ideas and offering up positive suggestions. Many thanks to every staff member in the ICT school, unfortunately too numerous to name one by one. However, I really appreciate them for all kinds of help with my research work as well as life in general, always encouraging me and sharing all kinds of things with me. Thanks too to every classmate in my office, the days working with them have been happy and fulfilling; they have added gorgeous colours to my monotonous and boring scientific research life. They have all shared my worries when my research or workload has proved most difficult, cheering me up when in a low mood and helping and encouraging me to extract every little drop from life – the joy and laughter we have felt when together has been unforgettable.

I also would like to thank my lover, as without his help, understanding, tolerance and support, I believe that the three years of doctoral life would be very different. While working by himself, he has helped me take care of our kids and all the affairs of my family. Despite our professional backgrounds being so different, he has still assisted me with clarifying my train of thought and sometimes in finding a breakthrough in the writing process of my paper; indeed, he has listened patiently and repeatedly about my academia travails. To conclude, I wish to offer a dedication to my beloved parents and in-laws; their silent support behind my back is the driving force for my progress. Here, I wish them all the best!

This research was supported by an Australian Government Research Training Program (RTP) Scholarship.

TABLE OF CONTENTS

List of Figures

List of Tables

Chapter 1. Introduction

Considering the first chapter, we introduce the background of blockchain-based conversation systems and state the research problems of the study. Furthermore, we outline the research questions and objectives of this study, and present the adopted methodology architecture and highlight the contributions of the presented work.

1.1 Research Background

Conversation systems are one of the core technologies in the field of artificial intelligence, and they act as a new harmonious human-computer interaction (Hu et al. 2013), which has a wide range of conversational services in the industry and today's intelligent life, (such as smart homes, companion robots, intelligent customer services, educational chatbots).

Conversation systems can be classified into two types: chit-chat (open-domain) systems and task-oriented (closed-domain) systems. Appendix A provides the theoretical background of these two types of conversation systems with typical examples. Irrespective of the task-oriented conversation system or chit-chat, both of them have five main components as shown in Figure 1.1. These include: automatic speech recognition (ASR), natural language understanding (NLU), conversation management, natural language generation (NLG) and text-to-speech (TTS) (Wen et al. 2016). Automatic speech recognition is utilized to convert speech signals to text. This occurs only in spoken conversation systems (Flores et al. 1988). Natural language understanding is utilized to interpret texts to obtain semantic representations that can be used by conversation management (Arora et al. 2013), whereas nature language generation involves constructing and mapping semantic frames into natural language (Perera & Nand 2017). Conversation management manages all aspects of the conversation content, and it usually needs to interact with external software such as a knowledge base. Considering these five main components, conversation management is the most important part, which controls the entire model of conversation (Wen et al. 2016) (Traum & Larsson 2003). Good and bad conversation management is directly related to the quality of conversation

content and degree of user satisfaction.

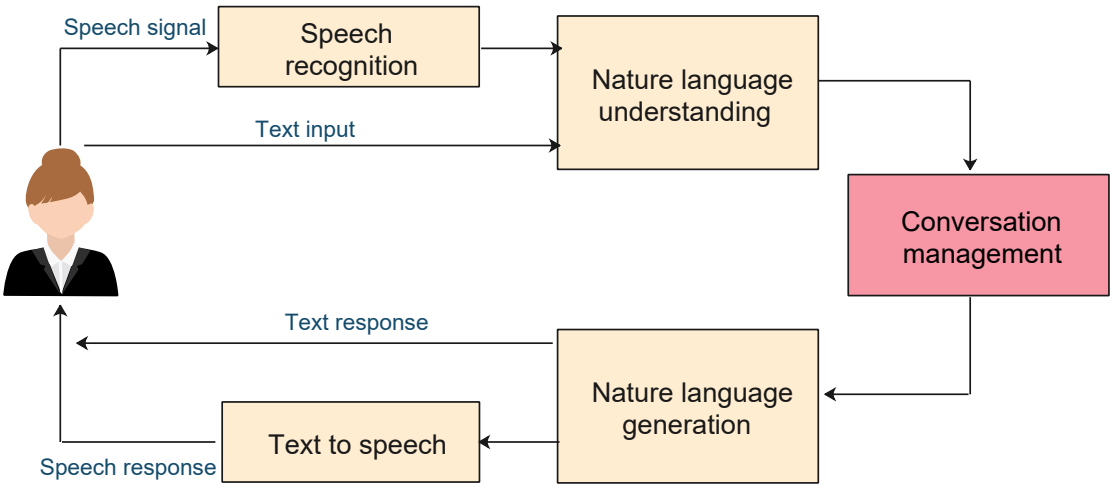


Figure 1.1: Basic framework of conversation system.

Recent studies on conversation management have demonstrated that conversation management with multi-expert groups is an efficient and qualified way to provide various conversational services (Papangelis et al. 2019, Black & Hunter 2009). There are three major security areas considered for managing multiple experts in conversation systems: incentive scheme, content tampering and privacy problems (Tamjidyamcholo et al. 2014, Safa et al. 2016).

Incentive scheme: To support multiple experts sharing and maintaining the various conversation contents, we need to provide trust and fair incentive scheme based on human motivation. The existing incentive schemes used in conversation systems include: gamification, recognition, and rewards. (Mmbaga 2018, Lai et al. 2019, Nian et al. 2020) These are always determined by administrators or third-parties, which are relied on certain authorities, and the results may not be trusted by participants.

Contents tampering: The incentives are calculated based on the contributions of participating experts; therefore, it is necessary to assign their contributions and keep those information secure and identified clearly. Existing secure content management in conversation systems such as proactive identification(Bertino et al. 2006), smart filtering of spam(Upadhyaya et al. 2011), keyword identification (Jiang et al. 2020) and centralized management of security administration(Buszta 2019) can provide audit trails using stored log files if there are any tampering contents. However it can still be easily erased or altered without trace.

Privacy problems: A user's identical information and some task solutions such as medical diagnosis

during conversational services is highly private or highly sensitive, which is the privacy aspect. Several studies have proposed solutions by providing privacy management such as security-assertion markup and managing access control, identity (Upadhyaya et al. 2011), digital rights management (Yan & Zhi 2005), privacy-preserving data mining (Aldeen et al. 2015), and circles of trust (Merminod et al. 2012). However, such privacy management is always based on centralized server, which still has the darkest secrets of data privacy.

It can be observed that the existing security technologies used in conversation systems can solve a certain aspect of the above problems; nonetheless, there has not been sufficient and efficient security mechanisms in guaranteeing all the above three issues. Recent developments and applications of distributed ledgers(such as blockchain) have indicated how to alleviate such issues, where incentives are based on consensus protocols, and each transaction is verified in a decentralized manner before it is recorded, preventing the occurrence of any illegitimate transactions. Therefore, it is necessary to build an efficient security mechanism to achieve trustworthy conversational services, and blockchain technology can be a solution to solve the current challenges.

1.2 Research Problems

Presently, many blockchain-based applications open doors; nonetheless, there is no unique solution for all the possible applications which encourages the development of new work. Considering the scope of this study, we address the three challenges described in Section **??**. The research problems of this study focus on the following two points:

1) Blockchain integration into a multi-expert conversation system lacks a solution that addresses the current challenges of conversation systems. Furthermore, various conversation scenarios may have different authority and permissions to manage (Nguyen & Wobcke 2005, Mourao et al. 2004). Thus, it is difficult to use the traditional single blockchain structures to manage them (Chauhan et al. 2018, Zheng et al. 2018).

2) Consensus and incentive performance are of great practical significance for effective conversation interactions among multiple expert. Currently, most consensus algorithms are simple and require long verification times even though the number of nodes in the network is relatively small (Mingxiao et al. 2017, Bach et al. 2018). In addition, the current incentive scheme used in blockchain platforms is always based on digital specie reward incentive consensus (DSRI) (Kratz & Strasser 2014), which

is used to motivate the participants using blockchain tokens. The token can be considered as a stock, currency, or goods. Using tokens can decrease the fund obligation of service providers (do not need to issue national currency immediately), and timely incentives can mobilize productivity (Xu et al. 2019, Darking et al. 2008). However, the DSRI model is imperfect; the main problem is that it attempts to express the behaviors of participants in the form of currency; however, expert behaviors can be affected by various types of costs and returns rather than only currency. In addition, the security of the DSRI will decrease with a reduction in the number of blockchain tokens.

In summary, we can observe that blockchain integration into a conversation system is still in an early prototype stage, and many essential characteristics need to be conducted and improved.

1.3 Research Questions

Based on the above analysis, the overall research questions explored in this study include the followings:

RQ: How can blockchain technology be integrated into conversation systems to provide security and trustworthy conversational services?

To comprehensively answer this research question, four sub-questions have been investigated in this study.

SRQ1: How are security aspects of blockchain Technology intended for conversation systems identified?

The first research sub-question is related to statistics of the current situation of conversation systems and conduct the vision of building blockchain-based conversation system.

SRQ2: How a suitable blockchain platform used for conversation systems selected?

The second research sub-question is related to identifying the key requirements of the conversation system and building a decision model for blockchain applicability to select the best fitting blockchain platform for further design and implementation.

SRQ3: How is the blockchain structure for conversation contents storage designed?

The third research sub-question is related to increasing flexibility and effectiveness in processing multiple conversation scenarios concurrently from different domains.

SRQ4: How is consensus and incentive mechanism for sharing and maintaining conversation

scenarios designed?

The fourth research sub-question is related to performing inter and intrachain consensus algorithm and incentive scheme by achieving trusted validation for conversation systems in a secure manner.

Table 1.1: Hierarchy of research questions for this thesis

SRQ1: How are security aspects of blockchain Technology intended for conversation systems identified?	RQ1.1 What is research landscape of blockchain technology in the conversation system?
	RQ1.2 What are key requirements of building a decentralized conversation system based on the blockchain technology to reach its fulfill potential?
SRQ2: How a suitable blockchain platform used for conversation systems selected?	RQ2.1 What indexes (such as centralization degree, resource usage, etc.) should be considered to evaluate different platforms?
	RQ2.2 How can the applicability evaluation model be built?
SRQ3: How is the blockchain structure for conversation contents storage designed?	RQ3.1 What should be the block structure and content?
	RQ3.2 How are the linking and validation mechanisms between chains designed?
	RQ3.3 What metrics will be used to assess the efficiency of the proposed blockchain structure?
SRQ4: How is consensus and incentive mechanism for sharing and maintaining conversation scenarios designed?	RQ4.1 Which consensus protocol is suitable for the designed blockchain structure?
	RQ4.2 Which incentive scheme is suitable for conversation system?
	RQ4.3 What metrics will be used to assess the efficiency of the proposed protocol?

1.4 Research Objectives

The objectives of this study include the followings:

1) The study aims to provide a review of the current situation of blockchain-based conversation systems to understand the challenges of conversational services and benefits of integrating with blockchain technology.

2) The study also aims to perform the requirement analysis of the conversation system and build a decision model to identify the best fitting blockchain platform for conversation systems by comparing the main blockchain platforms.

3) We also aim to design and implement a master-slave blockchain structure for conversation content storage and contribution verification to improve the security and efficiency of multi-domain conversation interactions.

4) Moreover, the study aims to design and implement a new hybrid consensus algorithm and incentive scheme for contributions and rewards verification during conversation contents sharing and maintaining based on the proposed master-slave blockchain structure.

5) We aim to achieve an efficient execution of smart contract-enabled decentralized knowledge fusion in conversation systems based on the proposed blockchain structure, consensus protocol, incentive scheme using multiple case studies to demonstrate the feasibility and effectiveness.

1.5 Research Methodology

To answer the outlined research questions and to achieve the objectives of this study, the overall architectural research diagram is shown in Figure 1.2. The methodological approach adopted to answer these research questions is based on the following steps:

Survey and statistical approach: considering any domain, a literature review to formulate the problem being studied is the most popular methodological approach. In this study, we comprehensively reviewed blockchain-based conversation systems. This approach focuses on the current situation analysis.

Decision-making approach: to apply the blockchain to a conversation system, blockchain applicability needs to be considered to make decisions using multiple criteria to formulate the problem as a hierarchical process. Decision-making involves modeling decision alternatives, model conduction, evaluation and verification of consistency. In this study, we utilized decision-making technique to provide blockchain applicability based on the selecting of the blockchain platform that best fits conversation system requirements.

Blockchain-driven approach to conversation management: to design and integrate a blockchain-driven approach, we proposed a design and developed decentralized conversation system based on the blockchain concept. The approach starts with a blockchain structural design, implementation, the essential

consensus protocol based on the proposed blockchain structure, and an incentive scheme among multiple experts. The blockchain-driven approach based on the above concepts can provide a novel framework for the realization of a decentralized conversation system.

Usability of the evaluation approach for blockchain-based conversation system: The necessary positive analysis include: smart contract enabled application design, data source selection and performance indexes for selection and evaluation analysis. Considering the evaluation analysis, we use quantitative methods to evaluate the proposed blockchain-based conversation system from many aspects.

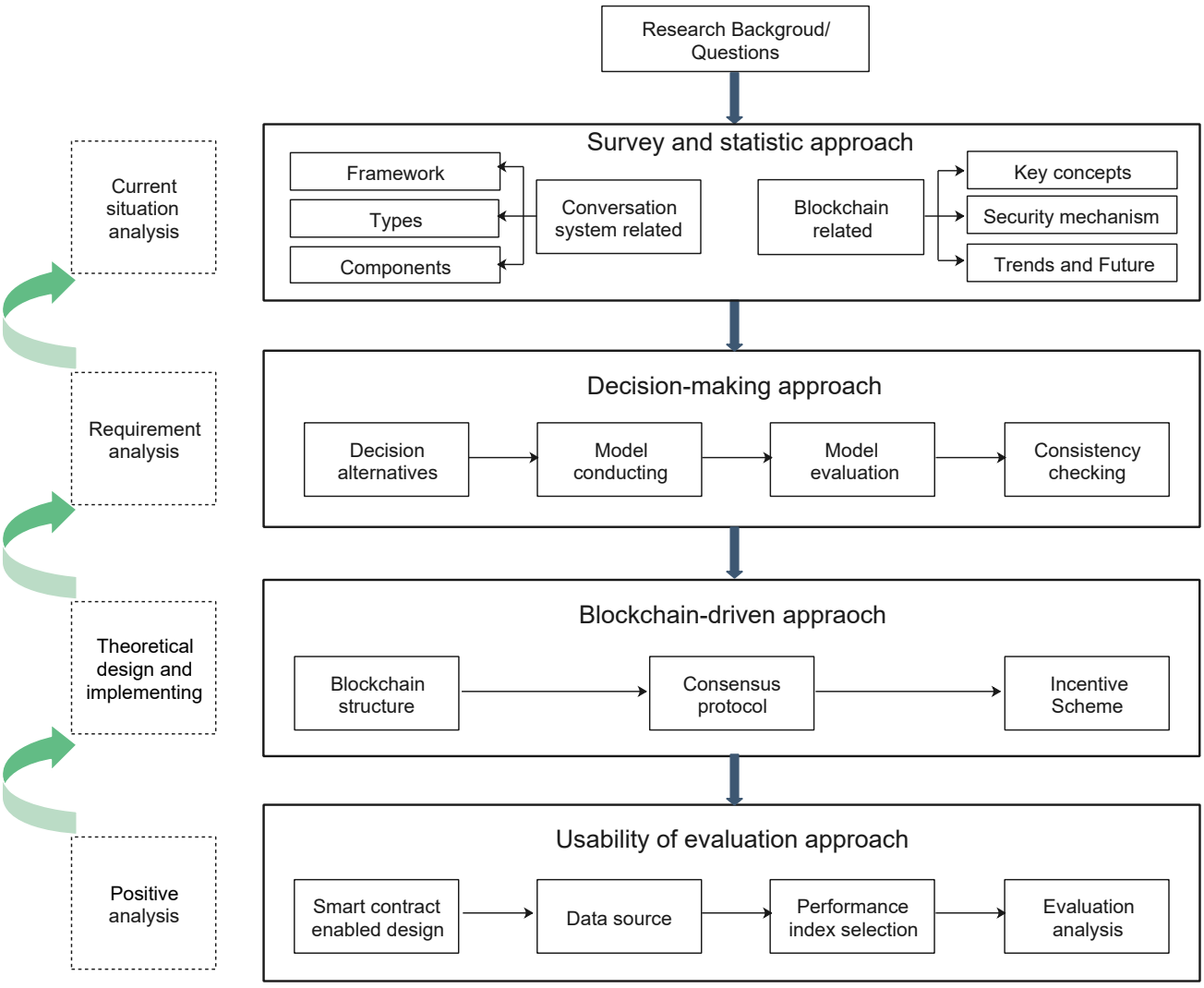


Figure 1.2: Architectural diagram of the approach.

1.6 Thesis Significance and Contribution

1.6.1 Significance of the study

This doctoral study addresses the following challenge: How to integrate blockchain technology into conversation management to improve security and scalability of conversation interactions? By integrating blockchain into conversation systems, the concept of conversational services is changed, since there is no single service provider but a set of participants that take on and share the tasks needed to run the conversation system. This has several remarkable consequences.

In terms of an incentive scheme among multiple experts, the benefit of this integration is the possibility of making trustworthy incentive decisions since the blockchain can ensure that all participants in a decentralised network share identical content and gain consensus. This assurance can allow the system to reach an agreement over the whole network and to have global collaboration between the conversation experts.

Regarding conversation content storage and authenticity, the usage of blockchain technology is useful in treating this problem by providing a reliable peer-to-peer communication with security and traceable measures over a trustworthy network, then conversations will be clearly identified in order to evaluate their contributions during sharing and maintenance.

Moving from a centralised conversation system to a decentralised system also means that supporting different conversation domains becomes possible, as the sharing of conversation scenarios and the exchange of values during complex transactions can simply utilise blockchain smart contracts. This has excellent scriptable programmability and would increase the fundamental baseline extensibility to support different types of conversation services.

1.6.2 Thesis contribution

This study makes the following concrete contributions:

1) It presents a comprehensive review of the integration of blockchain technology to secure conversation systems and describes the vision of building a blockchain-based architecture to guide further design and implementation of decentralized protocols. It captures the research landscape of blockchain technology in the conversation system and highlights the key requirements for building a

decentralized conversation system based on the blockchain technology. This contribution achieves the first objective, which is detailed in Chapter 2.

2) It clearly identifies the requirement analysis of the conversation system and presents a decision model for identifying the best fitting blockchain platform for conversation systems, which is the foundation of subsequent design and development. It can be utilized by researchers to identify the best fitting blockchain platform to support further implementation and support more reliable conversational services. This contribution achieves the second objective, which is detailed in Chapter 3.

3) A novel master–slave chain model for a conversation system has been designed and applied to process multiple conversation interactions from different domains concurrently. The master-slave blockchain structure is utilized for conversation content verification and storage. This contribution achieved the third objective and is detailed in Chapter 4.

4) A new hybrid consensus algorithm is proposed for master–slave chains to verify the feasibility and effectiveness of the proposed approach. An incentive scheme is designed to generate both economic and non-economic rewards for all the nodes participating in the consensus process. The proposed hybrid algorithm was used for collaborative sharing and maintenance validation. This contribution achieves the fourth objective and is detailed in Chapter 5.

5) A smart contract-enabled knowledge fusion application is implemented for decentralized conversation system based on the proposed blockchain structure, consensus protocol, and incentive scheme. And multiple case studies are utilized to evaluate the feasibility and effectiveness of the blockchain-based decentralized conversation system. This contribution achieves the fifth objective and is detailed in Chapter 6.

1.7 Thesis Organization

The thesis organization is outlined illustrating the overall structure and main contents of each chapter.

Chapter 1: contains the research background, considering the challenges of the current conversation system and the blockchain adoption. It also presents research problem statements and identify the research questions as well as objectives, outlines a summary of research methodology and states the main contributions.

Chapter 2 presents the vision of building a blockchain-based conversation system architecture based on the current situation analysis. Thus, the key requirements for integrating blockchain technology into a conversation system are revealed.

Chapter 3 proposes a decision model for blockchain applicability into conversation system that is used to select the best fitting blockchain platform for a conversation system. Consequently, the selection of blockchain platforms for conversation system is modelled as a decision-making problem with formulated multiple criteria regarding the identified requirement analysis and design aims. This can be easily scalable and can be updated to fit the new blockchain platform in the future market.

Chapter 4 investigates the problems of blockchain interoperability for conversation systems and proposes a novel master-slave blockchain structure to efficiently process multiple conversation interactions concurrently from different domains.

Chapter5 proposes a new hybrid consensus algorithm with an incentive scheme containing both economic and non-economic rewards, which is suitable for our designed master–slave chain. In this chapter, a consensus based on reputation-driven voting is utilized for intra-chain verification, where a dynamic construction strategy is used to select the master nodes for inter-chain authentication. Furthermore, an incentive scheme is designed to generate both economic and non-economic rewards for all the nodes participating in the proposed consensus process.

Subsequently, Chapter 6 further designes the knowledge fusion application in a conversation system based on the proposed blockchain-based decentralized mechanism and implements this through three case studies to verify the feasibility and effectiveness of the proposed approach. In this chapter, we utilized the proposed blockchain structure, consensus protocol and incentive scheme for knowledge fusion in a conversation system to perform the positive analysis.

Finally, Chapter 7 concludes the study with a summary of the significant research outcomes and highlights the future research works.

1.8 Publication Record

The work presented in this thesis has been partially or completely published in the following set of publications presented in chronological order:

- Wenli Yang, Saurabh Garg, Ali Raza, David Herbert, Byeong Kang, "Blockchain: trends and

future", Proceedings of the 15th Pacific Rim Knowledge Acquisition Workshop: Knowledge Management and Acquisition for Intelligent Systems, 2018, pp: 201-210. Chapter 2 is derived from this publication.

- Wenli Yang, Erfan Aghasian, Saurabh Garg, David Herbert, Leandro Disiuta, Byeong Kang, " A survey on blockchain-based internet service architecture: Requirements, challenges, trends, and future", IEEE Access 7, 2019, 75845-75872. Chapter 2 is derived from this publication.

- Wenli Yang, Saurabh Garg, Zhiqiang Huang, Byeong Kang, "A decision model for blockchain applicability into Knowledge-Based conversation system ", Knowledge-Based Systems, 2020, 220: 106791. Chapter 3 is derived from this publication.

- Wenli Yang, Saurabh Garg, Zhiqiang Huang, Byeong Kang, " Hybrid consensus algorithm for master–slave blockchain in multi-domain conversation system", Expert systems with Applications (under review). Chapter 4 and 5 are derived from this publication.

- Wenli Yang, Saurabh Garg, Bai Quan, Byeong Kang, "Blockchain-based Decentralized Mechanism for Knowledge Fusion in multi-expert Conversation system.", Expert systems with Applications (under review). Chapter 6 is derived from this publication.

Further published work, which is outside the research scope of this thesis:

- Wenli Yang, David Hebert, Sunwoo Kim, Byeong Kang, "MCRDR knowledge-based 3D dialogue simulation in clinical training and assessment", Journal of medical systems, 2019, 43.7: 1-21.

- Wenli Yang, Nanqi Yuan, Winyu Chinthammit, Byeong Kang. "A distributed case-and project-based learning to design 3D lab on electronic engineering education", Computer Applications in Engineering Education, 2019, 27.2: 430-451.

- Wenli Yang, Shuangshuang Fan, Shuxiang Xu, Peter King, Byeong Kang, Eonjoo Kim, "Autonomous underwater vehicle navigation using sonar image matching based on convolutional neural network", IFAC Conference on Control Applications in Marine Systems, Robotics, and Vehicles, 2019, 52.21: 156-162.

- Mira Park, Wenli Yang, Zehong Cao, Byeong Kang, Damian Connor, Mary-Anne Lea., "Marine vertebrate predator detection and recognition in underwater videos by region convolutional neural network", In Pacific Rim Knowledge Acquisition Workshop, 2019, pp: 66-80.

- Jingqiang Yuan, Weizhong Chen, Xianjun Tan, Wenli Yang, Diansen Yang, Hongdan Yu, Bo Zhou, Baohua Yang, "Study on the Permeability Characteristics of Foamed Concrete Using a Pore-Scale Model from X-Ray Microcomputed Tomography Image Reconstruction and Numerical Simulation", Journal of Materials in Civil Engineering, 2021, 33 (6): 04021117.
- Wenli Yang, Mira Park, Xianghui Song, Sun Ling, Yameng Li, "Vehicle Detection Based on Cascade Deep Learning Method Using Deformed Oriented Bounding Box",Australasian Joint Conference on Artificial Intelligence 2021 (under review).

Chapter 2. Survey on blockchain-based conversation system

Conversation systems can refer to different kinds of communication over the Internet that offers data transmission between service requesters and providers, which is a typical Internet services, we call conversational service. The aim of the conversation system is to provide an efficient and trustworthy conversational service. Current conversational services allow us to access conversation interactions through the Internet; nevertheless, few studies focus on both fundamental security and effectiveness, especially when they involve different conversation domains with overloaded private information, which always happenes in conversation systems.

In this chapter, we explore the blockchain-based mechanism that aims to improve the security of the current conversation system. Furthermore, we provide a review to conceptualize the blockchain-based framework to build the vision of decentralized conversation system. Finally, we summarize the trends and challenges of blockchain technology that benefits a multitude of disciplines across the conversation system. This comprehensive survey aims to conduct the current situation analysis of developing a conversation system which can provide a trustworthy conversational service.

2.1 Introduction

Similar with many other internet services, the original conversational service architecture was to build a common decentralized network with equal participation, that communicated using peer to peer inter-connectivity without relying on a single computer (Braden et al. 1994). Another important consideration of the original conversational services plan was that computers must be interoperable among dissimilar systems, so that more devices could be a part of the network.

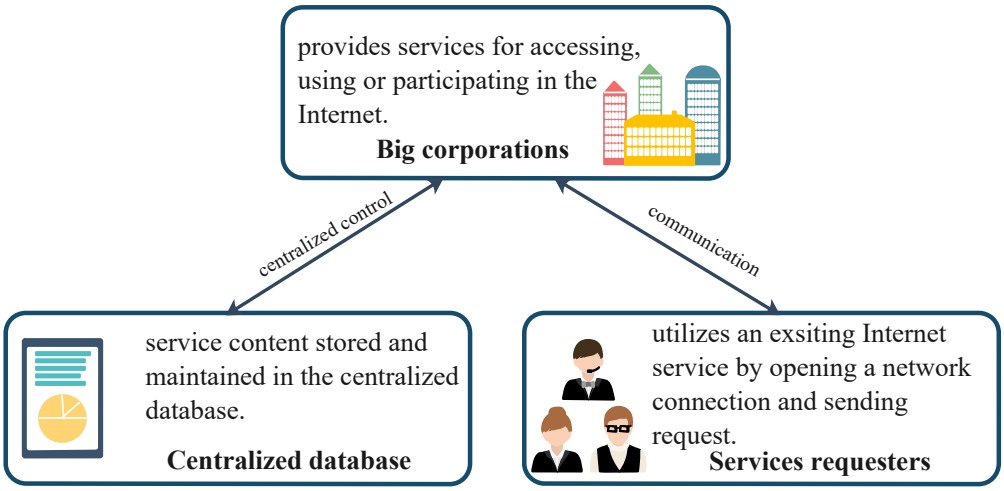
However, after the first dot-com bubble (Ljungqvist & Wilhelm Jr 2003), large corporations (such as Google and Amazon) realized that the largest value gained from this decentralized network involves

gathering, organizing, and monetizing information through centralized services. These companies therefore built their value by growing massive centralized databases using freely-obtainable private, personal data that is then deployed on the Internet, and these changes led to the conversational service architecture partially deviating from the original architectural intentions.

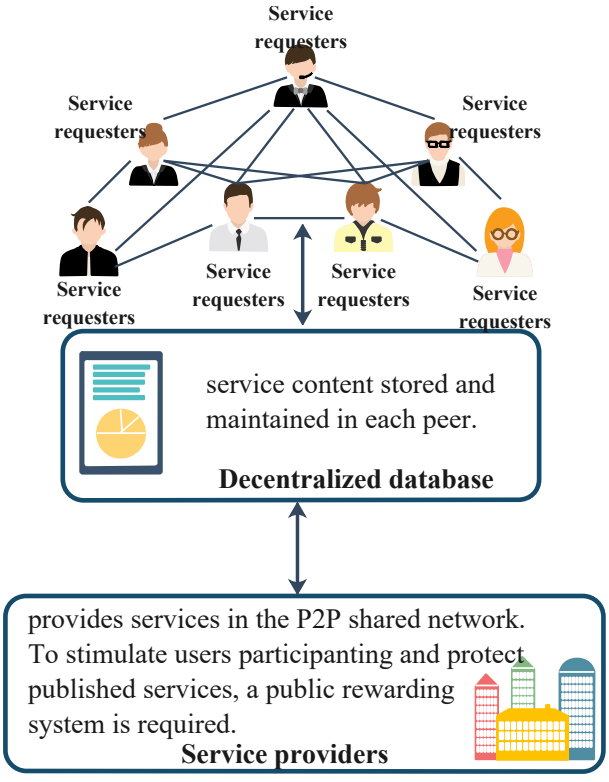
Today, the conversational service is physically decentralized, but it contains critical components for text processing, knowledge management, data storage that use large centralized services. The traditional conversational service consists of three groups of roles: service requesters, big corporations (service provider) and the centralized database (Figure 2.1). Service requesters are responsible for requesting services from service providers who provide various conversational services. Almost every service provider has its own data center, where it stores user data and runs its applications. As shown in Figure 2.1(a), it can be seen that as the public has a greater reliance on such services, it is of substantial fiscal benefit for the big corporations to keep their services maintained and remain proprietary.

However, such concentrated centralization has also created a growing number of issues (Van Schewick 2012). First, traditional conversational service architectures are vulnerable to denial of service, which makes the services unavailable, such as the global financial crisis (GFC) of 2008 (Nakamoto n.d.). Secondly, the majority of conversational services rely on the centralized database, which suffers from a single point of failure, as they provide attackers a single target to hack. For instance, when centralized services such as LinkedIn or Gmail Services fail, all the websites and applications that depend on them stop working. Third, users' identity information (e.g. name, email address and phone number) and task solutions are saved in a centralized database, which now may contain many aspects of concern to data privacy. Users can never tell about what goes on behind the walled gardens of centralized services. Therefore, they do not precisely know how much data these services collect about them and how that data is used. Furthermore, when a service requester and provider are in dispute, they need a trustworthy network to give a subjective arbitration, which may lead to a behavior known as 'error-reporting'. In short, it can be seen that the existing conversational service implementations achieve information transmission and sharing in a decentralized manner, but there has not been sufficient scrutiny and action in guaranteeing transactional trust and the exchange of wealth or value across the Internet.

Therefore, building a trustworthy conversation system is a very important and fundamental task. There have been many research topics to deal with part of the above mentioned issues in conversational services. These topics are mainly related to attack detection and prevention, failing with single-point solutions and privacy protection. For example, data anonymization (Zhou et al. 2008, Ghinita et al. 2009), differential privacy (Xiao et al. 2011, Kairouz et al. 2014) and encryption schemes (He et al. 2014,



(a) Centralized conversational services



(b) Decentralized conversational services

Figure 2.1: The old and new way of conversational service

Peikert 2014, Heuer et al. 2016) are proposed to protect personal data privacy. Reputation-based security mechanisms are designed to identify and predict transaction safety based on overall use and reputation over a wide community of users. Distributed architectures are proposed to address the single point of failure problem. However, at present, none of the existing work has solved all issues simultaneously. Therefore, our research is motivated by how to design a decentralized framework with distributed data verification and security, where blockchain technology potentially fulfills this purpose as shown in Figure 2.1(b).

Blockchain is a relatively new platform technology, which is widely known and it was developed primarily to use with Bitcoin cryptocurrency (Yli-Huumo et al. 2016). Blockchain is based on decentralized networking and one of its main characteristics is to guarantee the safety and integrity of data. The technology is robust and all participant nodes provide resources in a fair manner, which alleviates many-to-one traffic flow bottlenecks. This technique decreases traffic delays and defeats the errors due to a single point of failure (Dorri et al. 2016, Conoscenti et al. 2016).

To address security and trust concerns, Hart claims that a network framework cannot be based on a single entity to manage the network's infrastructure. Instead it requires peer to peer (P2P) resource management (Hari & Lakshman 2016). Therefore, blockchain would be an ideal solution to secure the conversation system in addition to the various services layered upon it. This would increase the fundamental baseline security and as blockchain has excellent extensibility features such as scriptable programmability, and it supports new types of layered conversational services.

Since 2009 to now, blockchain has attracted a considerable amount of attention in applied fields ranging from Bitcoin to financial services, supply-chain management, Internet of things and so on. Many researchers think 'conversational service + Blockchain' represents an ideal solution to build a new decentralized architecture with value at a low cost (see Appendices B), the related research works are limited to some specific internet services rather than conversational service, thus for conversation system, it still needs to design and implement of decentralized protocols.

2.2 Existing Conversational Service Architectures

In conversation system, conversational service architectures typically cover the basic communication between heterogeneous networks that may differ internally in terms of hardware, software, or technical design. Building a secure, layered service architecture is vitally important to ensure that all commercial requirements as well as the user's demands are achieved, but not at the expense of a robust and trusted

security model. Software security mechanisms have evolved from a single-tier architecture, to two-tier architecture, and to the current multi-tier architecture (Van Schewick 2012, Barais et al. 2008) (refer to table 2.1). Through this evolution, it can easily be seen that the existing security mechanisms are centralized or have a locally centralized architecture.

Single-tier service architecture: this architecture is used for simple conversational services in which the user interface and data access are combined into one single program integrated into a single platform (Melis et al. 2016). In this architecture, the control and data plane share the same host server (figure 2.2). This architecture is very easy to implement in the early stages of service deployment, however, it is unable to satisfy complex applications as it introduces a single point of processing (bottlenecks) as well as a single point of failure. Also, the security mechanisms for single-tier services consider authentication and authorization. Authentication is used to verify the identity of a user, while authorization manages what a user can or cannot access, focusing on permissions.

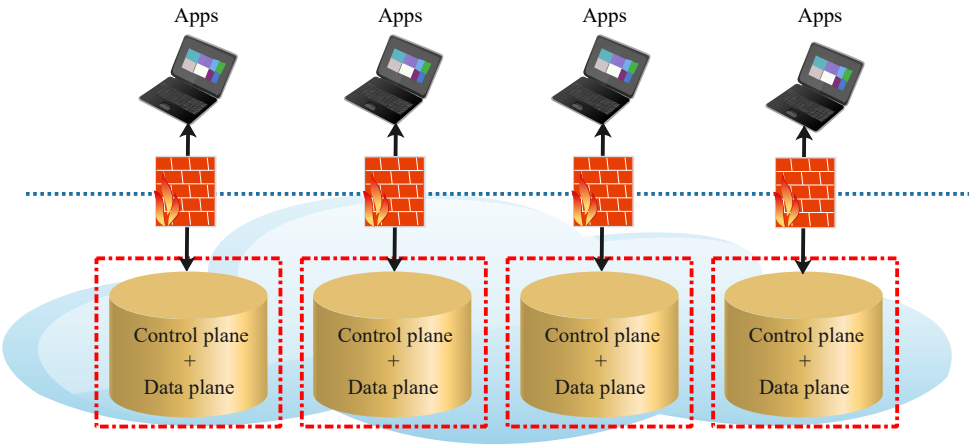


Figure 2.2: Single-tier service architecture

Two-tier service architecture: this architecture separates the control plane acting as an interface for a single host machine, from the data plane which is used to store data on another server (Terzis et al. 1999). Separating these two components into different locations represents a two-tier architecture (as depicted in Figure 2.3). Although the database server is separated from the single server deployed in single-tier architecture, servers still remain a potential single point of failure within the two-tier service architecture.

Multi-tier service architecture: this architecture divides different components into multiple planes according to their functions. Each plane runs on a separated server (Urgaonkar et al. 2005). Multi-tier can be classified into two main types depending on the control mechanism: distributed and centralized control (as shown in figures 2.4(a) and 2.4(b)). A distributed control plane allocates control protocol

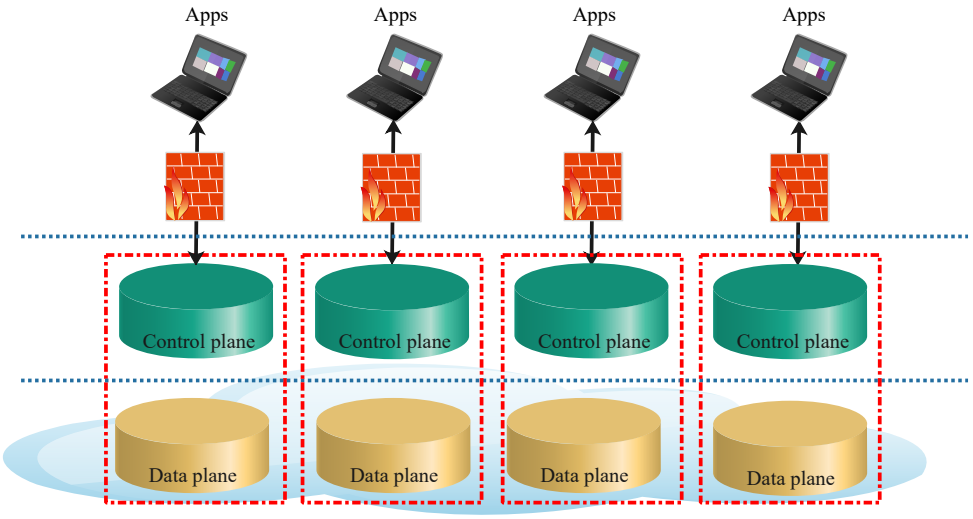


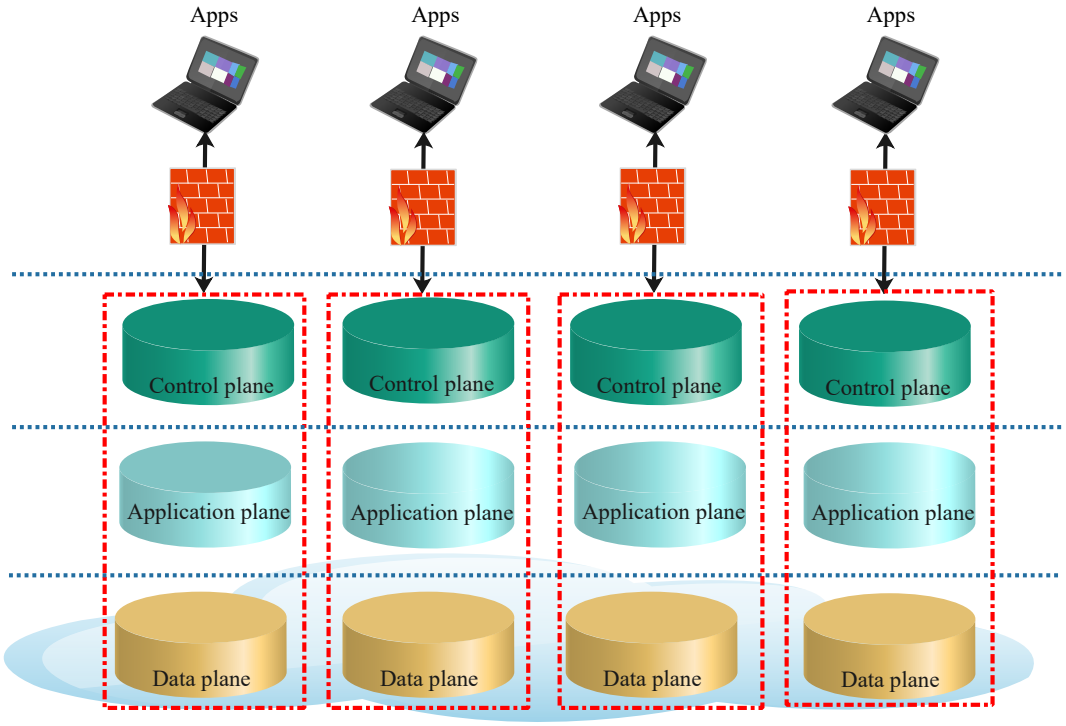
Figure 2.3: Two-tier service architecture

functions across multiple processor levels in the network, while a centralized control plane, like the SDN network architecture, aims to improve network performance in terms of providing centralized network management capabilities (Hu et al. 2014). Both methods provide compartmentalization and avoid a single point of failure. Although the implementation of a multi-tier service architecture could help to enhance system security, it still uses several controllers to concentrate on published conversational services or applications.

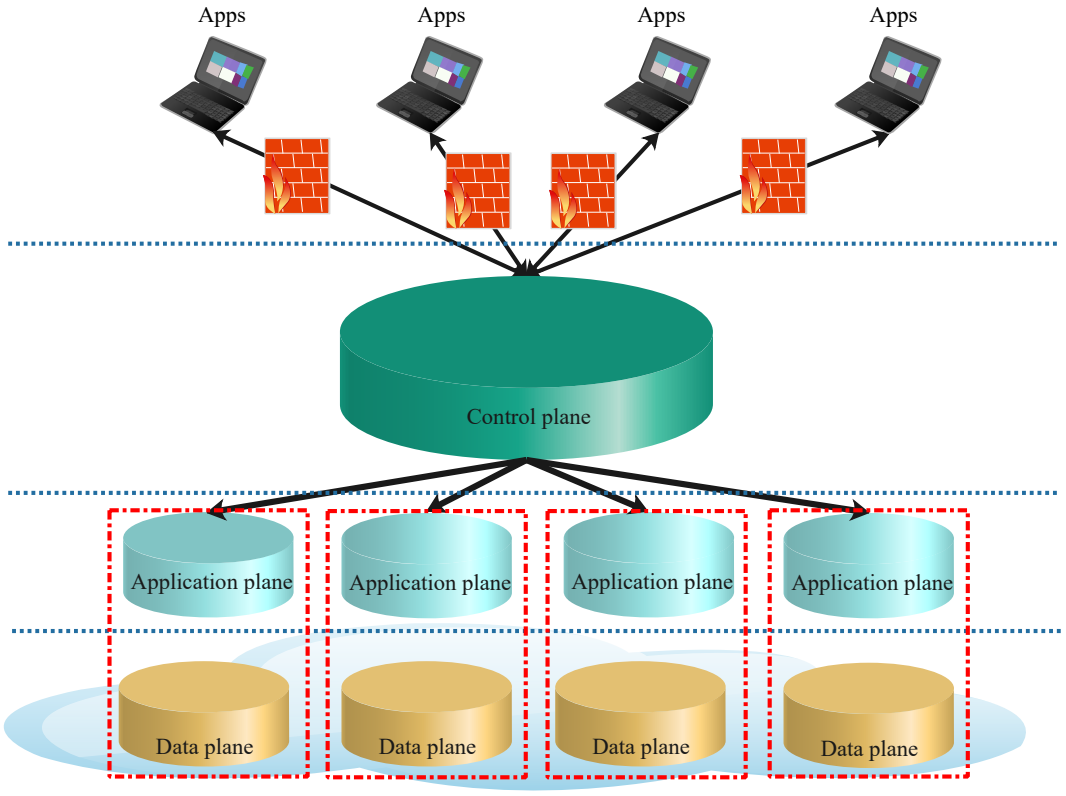
Existing conversational service architectures can utilize high speed data transmission and enable the efficient use of resources. However, as shown in Table 2.1, there are several limitations and challenges that need to be addressed, especially with regards to application security issues (Barrera et al. 2017). Some of these issues are:

Data obtained from non-verified sources: Currently, the huge amount of power which services such as Google and Facebook have as reliable sources of information, has turned them into gatekeepers of information - the public can only believe them based on trust. For example, if Google wants to express some fake and misleading content to the users, there is virtually no method to stop them. The recent anecdotal swing of the 2016 USA federal election to the Republican Party due to the spread of fake news via trusted social network platforms like Facebook and Twitter highlighted that the trust can be misplaced (Allcott & Gentzkow 2017).

Many sources rely on their own data: Almost every Internet company or business has its own data centre, where it stores user data and runs its own applications. This requires some serious security, as they are large and obvious targets for hackers attempting to steal sensitive data. But, due to self-reliance, when centralized services such as LinkedIn or Gmail fail, all associated applications that depend on them



(a) Decentralized control



(b) Centralized control

Figure 2.4: Multi-tier service architecture

Table 2.1: Comparison of existing Internet service architectures

Evolution	Single-tier	Two-tier	Multi-tier	
			Distributed control	Centralized control
Typical application	Local desktop database e.g. Microsoft Access with local presentation services.	Desktop applications, e.g. spreadsheet and word processing via file sharing.	Almost all web applications user a three or Multi-tier architecture.	
Points of failure or maintenance	Easy to maintain as there is only a single point of failure.	Easy to maintain and modification is relatively easy.	More difficult to maintain.	Easy to maintain and deploy.
Easy of development /Creation	Simple to create. Standardized separation of data and presentation e.g. MVC (model, view, controller) framework assists with development.	Slightly more complex to create and develop issues such as contention and concurrency need to be considered.	More complex, need to preestablish lower plane details such as data sharing and transmission capabilities.	Fast creation. Apply to everywhere with the single framework.
Network performance	Lower relative performance, and difficult to support large and complex network traffic access patterns.	Communication is faster than single-tier, but the server request response rate is a bottleneck, as a result it can cause data integrity issues.	High performance, but network operations cannot be easily reprogrammed or retasked.	Highest performance without a device-centric configuration on each location.
Scalability	Very Poor.	Still poor scalability, application performance will be degraded with increasing user count.	Each tier can scale horizontally, but at the expense of increasing complexity or effort.	Tiers (except the control plane) can scale horizontally.
Security	Locally centralized, rely on authentication and authorization between one server and users.	Locally centralized, similar with single-tier, if one server crashes, the corresponding application will stop.	Locally centralized, although client does not have direct access to the database, it still relies on authentication and authorization between servers and users.	Totally centralized, but will be highly unstable.
Key examples	(Manuel & Al-Ghamdi 2003) (Kambalyal 2010) (Yang et al. 2019)	(Liu et al. 2008) (Joe et al. 2016) (Yang et al. 2019)	(Edlund & Hjalmarsson 2005) (Tang & Daoutidis 2017) (Budzianowski et al. 2018)	(Kruijff-Korbayová et al. 2011) (Laihonen & Mäntylä 2017) (Czyzewski et al. 2020)

are unavailable. This creates a very visible and widespread concern when such services fail.

Lack of security for private data: The existing conversational service architectures also involve privacy concern problems. Users are unaware of what occurs behind the walled gardens of centralized conversational services. In other words, users are not notified of how much of their private data is being gathered by these services and what purposes the data will be used for. Recent (May 2018) (Voigt & Von dem Bussche 2017) legislative changes in Europe with the introduction of the General Data Protection Regulation (GDPR) highlight the seriousness of the issue. Application service providers with clients in Europe scrambled, some seemingly at the last moment, to be compliant with the legislation. Unfortunately, such compliance did not necessarily extend to clients in other non-European countries,

and a universal, international regulatory protection is currently lacking.

The birth and development of blockchain aims to solve the security and trust problems faced by the current conversational services. It would remove single points of failure due to distributed ledgers. Blockchain would prevent single data storage based on peer-to-peer networking, as opposed to traditional client-server models. Blockchain would also enhance competition by avoiding lock-ins and giving users full control of their data (Pop et al. 2018).

2.3 Key Concepts of Blockchain

A blockchain is based on Distributed Ledger Technology (DLT) that is spread across several nodes or computing devices. It is assumed that these nodes do not fully trust each other as some may exhibit Byzantine (dishonest) behaviour. These nodes maintain a long chain of cryptographic hash-linked blocks where each modification or addition of a transaction is validated by the consensus of all nodes in the system. In one sense blockchain is similar to a traditional database requiring ACID properties (He et al. 2018) to be satisfied. The key difference is the 'distributed consensus' algorithm which decides whether a new block is valid and legitimate before any insertion can be done. Figure 8 shows the data structure of a blockchain whose basic concepts include:

- Transaction: an operation that caused a change of the block.
- Block: a container data structure, and a block is composed of a header and a long list of transactions.
- Digital trust elements: include crypto and hashing, which make the blocks linked and secured using cryptography.
  - Encryption using Public / Private Keys: is one of the core features of blockchain technology that plays an important role to keep the blockchain safe while ensuring the transactions safe and prevent from fraud. To ensure the security properties, public key cryptography is used in the blockchain. Each user in blockchain have two cryptographic keys such as public key and private key and keep them in the personal wallet. To send the transaction to another user in the blockchain network, a sender requests the public key of the receiver, while in response, the receiver sends their public key to the sender. Generally, the public key is known to everyone and used for identification purpose such as email id of any person. On the other hand, the private key is a secret key which is kept hide and used for authentication and encryption to secure the data. In addition to these functionalities, the private key is used to derive the public

key by performing some algorithmic computations which are further derived into the address of that user. Moreover, both public and private keys are used in combination to produce a signature of the transaction.

- Hash: is a digestive or compress function which is used to uniquely identify the data (Wang, Duan & Zhu 2018). The hash function always gives the unique output same as the fingerprints of a human. Once the block is created to store some transactions, its hash is created by calculating the stored transactions in them. The important feature of the hash function is that it immediately detects changes in the data by generating different hash value. Therefore, hash functions are used to achieve the data integrity to ensures that data is not altered in the communication.
- Hash Chains: In blockchain, chaining is a process of linking all the blocks in the form of a single chain. In the chaining process, the hash of each current block is calculated and stored in the header of the next block and this process continues till the last block added in the blockchain. Therefore, the hash is considered as an essential component to link all of the blocks. The advantage to link the blocks in this fashion is that if someone doing the changes in the data, it automatically changes the hash value of a current block that would be different than the hash stored in the next block. So, it is computationally hard for anyone to changes the hash values of all linking blocks.

• Distributed Consensus: is the way to resolve the conflicts between disputing parties when they interact seamlessly and participate in forming the blockchain after verification of transactions. Most of the nodes in the blockchain agree on one common point for the correctness of block. The advantage of the consensus mechanism is that it prevents the single users to control the whole blockchain system. It also ensures the maintenance of a single chain throughout the whole network. For example, the Figure 7 shows the work of one of the most popular consensus algorithms Proof-of-Work used in blockchain system.

As the above description, blockchain is a relatively new platform technology, which is based on decentralized transaction and data management. It can provide anonymity, safety and data integrity (Wurster et al. 2018). There is no need of a third-party organization to control the blockchain transactions, making this field a vast research area to deal with limitations and enhancements in the current knowledge fusion construction. Blockchain combines multiple technologies to ensure an immutable, irrevocable and traceable blockchain ledger (Viriyasitavat & Hoonsopon 2018). The basic architecture of blockchain comprises date, network, consensus, incentive, contract, and application layers (Syed et al. 2019) from

bottom to top as shown in Figure 2.5.

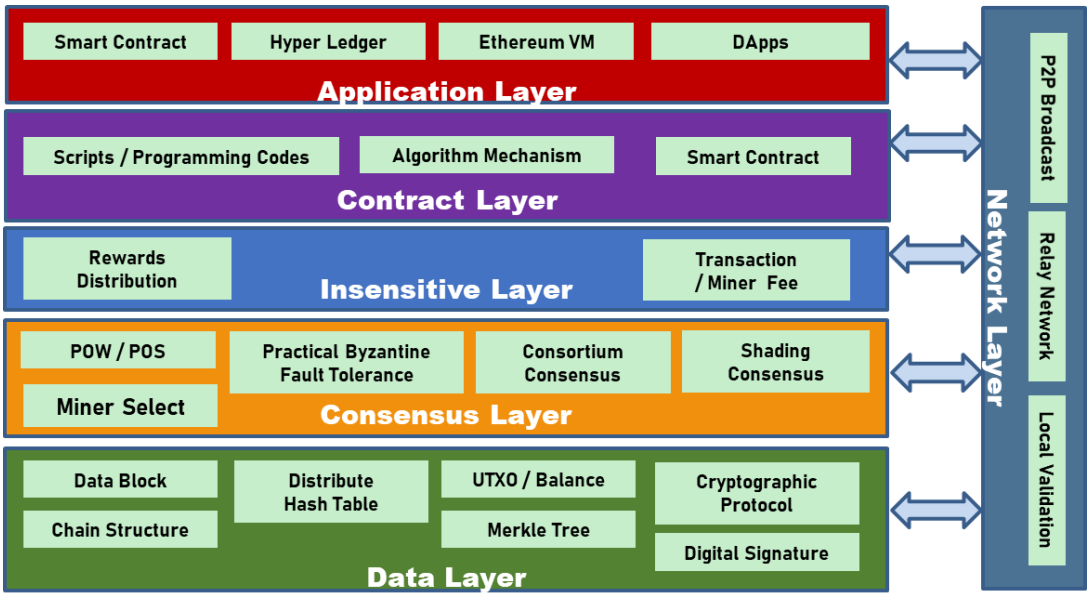


Figure 2.5: The blockchain Basic archiecture.

- Data layer: includes the underlying data blocks in a chain structure, related asymmetric public and private keys encryption, and timestamp technology.

- Network layer: includes P2P networking, data transmission, and data verification mechanisms. The P2P networking mechanism is early adopted in P2P download applications like Bitcoin, which means the blockchain technology has self-organizing network function. Propagate and broadcast transactions among nodes.

- Consensus layer: consists of various consensus algorithms for nodes on the network. Consensus algorithms are the core technology for blockchain, since it determines who keeps records, which would influence the security and reliability for the whole system.

- Incentive layer: integrates economic factors into the blockchain system, including issuance of economic incentives and allocation mechanism, which mainly occurs in public chain. And in private chain, it is not necessary to offer incentives, because the nodes, participating in the record-keeping, usually complete their competition off the chain forcibly or voluntarily.

- Contract layer: it is the basis for blockchain programmable feature, which are automatically executable and enforceable. No arbitrator or third party can influence or manipulate the execution of smart contracts, which means smart contracts can ensure each node in the network will execute the same contract and get the same result.

- Application layer: includes various application scenarios and use cases, for example, the blockchain applications built on Ethereum are deployed on this layer.

2.4 Blockchain VS Traditional Security mechanisms

Contrary to traditional security mechanisms for common conversational services, blockchain technology is based on decentralized transaction and data management which is able to provide anonymity, safety and data integrity (Wurster et al. 2018). There is no need for a third-party organization to control the blockchain transactions, making this field a vast area of research to deal with limitations and enhancements within the current conversational service architecture. Blockchain combines multiple technologies to ensure an immutable, irrevocable and traceable blockchain ledger (Viriyasitavat & Hoonsopon 2018). This section will discuss the security of blockchain technology through different planes in the conversational service architecture (data, control, network and application planes) compared with traditional centralized-based mechanism.

2.4.1 Data plane

The data plane manages the required data, such as data storage, sharing and retrieval. The main difference between a traditional database and the blockchain database is data structure. The most common data structure of traditional conversational services is a database table that in essence consists of a two-dimensional array indexed by a row and column value. Other data structures such as b-tree and a user-defined vector are also in common use. Traditional database management is operated by one or several controllers on the basis of a hierarchical data structure and have been principally secured against hackers over network security mechanisms like network-based intrusion detection systems (IDS) and firewalls. However, these security mechanisms are still high risk. For instance, if one table in the database is corrupted, the operation of the whole database is potentially compromised and the data access would be lost (Barrera et al. 2017). Even if appropriate maintenance processes are in place, data loss may still occur even after a rollback or table restoration. Unlike the traditional conversational services, blockchain is based on Distributed Ledger Technology (DLT) (Walport 2016), which is spread across several nodes or computing devices. Blockchain uses a chain data structure based on cryptography algorithms consisting of a transaction, block and a chain as shown in Figure 2.6. A transaction is an operation that causes a change to the whole ledger between nodes and a block is composed of a header and a long list of transactions. All nodes in the system maintain a long chain of blocks which are linked and

secured against tampering by the application of cryptography techniques. The composite structure "block (complete history) + chain (complete verification) = timestamp" provides an integrated and immutable database. This structure provides a better data integrity when compared with traditional services.

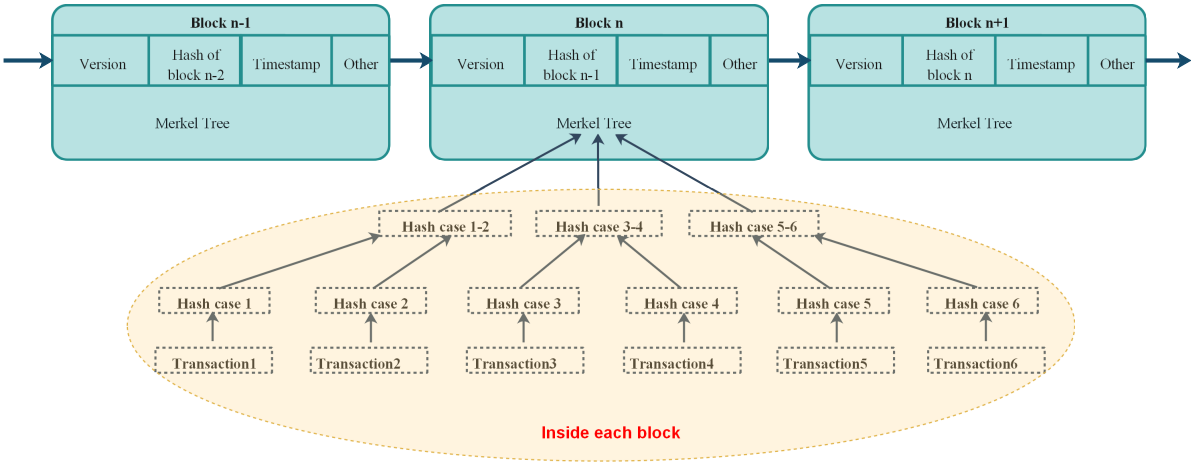
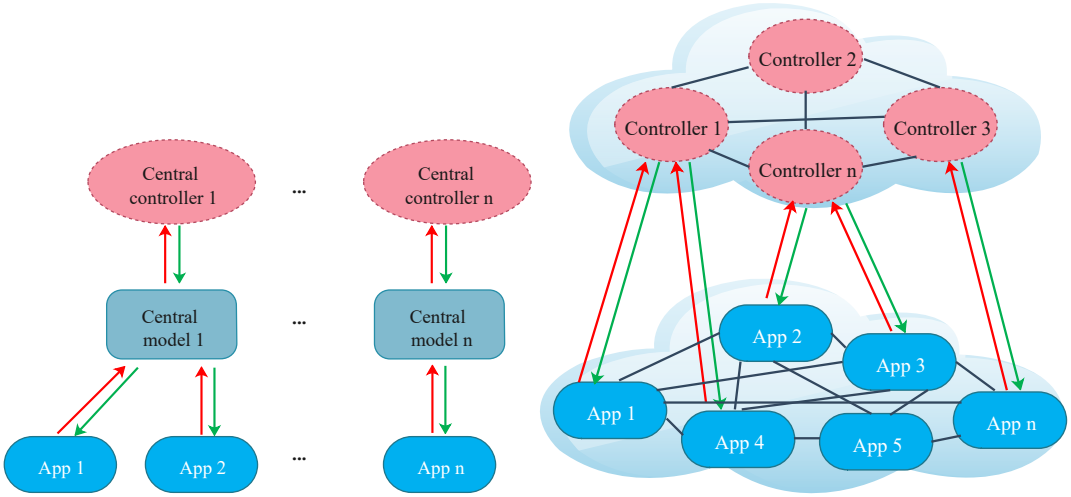


Figure 2.6: The basic data structure of blockchain.

2.4.2 Control plane

The control plane advertises and displays information related to services available on the Internet. The control protocols used in conversational services can be divided into three main types: centralized, distributed and decentralized models. Contemporary conversational services use a globally centralized controller or a locally-centralized controller to communicate with the data plane as well as the application plane. Centralized control is usually comprised of one device that deals with tasks such as I/O connectivity, motion control and so on (Ali et al. 2016).By using a centralized mechanism, administrators have the ability to effectively manage the traffic data from different locations. Since the control calculations are performed in the central device, the computing capacity demands have to be significantly higher with corresponding and security requirements which have to mitigate the associated risks. In order to overcome this, a distributed control structure was illustrated (Smaragdakis et al. 2014) using locations and facilities re-optimizing, which shows good scalability in simulations. Research has found that using a distributed model to provide conversational service could prevent service break resulting from the loss of networking or power (Gribble et al. 2000). Although the implementation of distributed control model focuses on allocating control protocol functions across multiple processor levels in the network, they had a centralized platform to provide services, which is not consistent with the requirement of building conversational services in a decentralized way. Decentralization is basically to distribute constraint and dominance from the central authority to peripheries in order to weaken the centralized

organizations' function with secured benefits (Bashir 2017, Zhaoyang et al. 2018, Pinyaphat 2018), which can makes use of the information exchanged between distributed controllers allocated within the control plane. This process can ease the access control and revocation within the system (Ali et al. 2016). The blockchain utilizes decentralized control as independent organizations or individuals are usually distributed geographically. The main advantage of decentralized control is that the presentation authority is delegated to the individual nodes throughout the network rather than limiting it to a few executive nodes. Figure 2.7 depicts a comparison between the centralized, distributed and decentralized control plane.



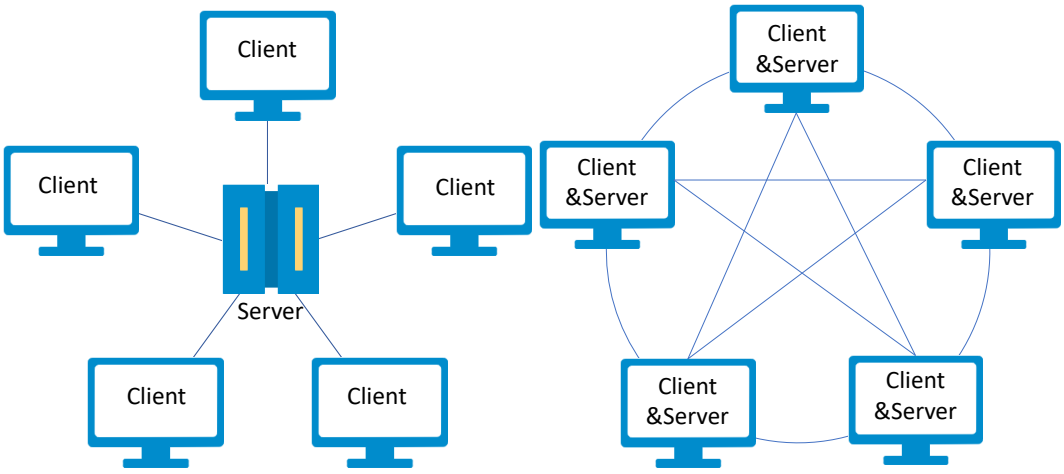
(a) Centralized or local centralized control (b) Decentralized control

Figure 2.7: Comparison between control plane

2.4.3 Network plane

Traditional conversational services use a client-server infrastructure. Each user acts as client and can query data that is stored on a centralized server. Since the centralized control is accountable for database administration, if the authority's security is compromised, the data can be modified or even deleted (Shahin et al. 2018). In contrast, blockchain is based on a peer-to-peer (P2P) network structure consisting of several decentralized peers. In terms of data integrity, blockchain defines a set of protocols, which verify each participating node in the network when a new transaction is created. Then, the new transaction record is integrated into the block only after the majority of nodes reach a verification consensus. Regarding data storage, blockchain is based on a distributed architecture, where each node has a backup of the whole ledger. This means that if a node is corrupted or in-accessible, the integrity of the database will not be affected. Hence, through the distributed transmission of data, record of transactions and

distributed storage, the entire architecture can be defined as decentralized in nature. This decentralized architecture improves the speed, flexibility and security by reorganizing the application service network, and it provides for a more efficient local control and execution capability of a service (Croman et al. 2016) (Figure 2.8).



(a) Centralized network (b) Decentralized network

Figure 2.8: Comparison between network plane

2.4.4 Application plane

Many Internet applications can be generally considered as centralized applications that focus processing in one host or in a cluster of coupled computers in a single location. For instance, the purchase process from EBay website can go through PayPal. PayPal is a typical centralized application that concentrates all transactions between the seller and buyer. If PayPal's data-centre or cluster is compromised, its transactional history and balances can no longer be trusted leading to further service disruption to those that rely on PayPal. Decentralized applications (Dapps) differ from centralized applications and are a type of software program on the Internet that are designed in a way that they are not being controlled by any single entity. In order to have an ideal service or blockchain application, there should be no human intervention in the operation which leads the formation of an autonomous organization that is decentralized. The autonomy can help to share the profit and the cost into the blocks (Cai et al. 2018). There are noticeable common features of Dapps which are completely hosted by peer-to-peer blockchain system:

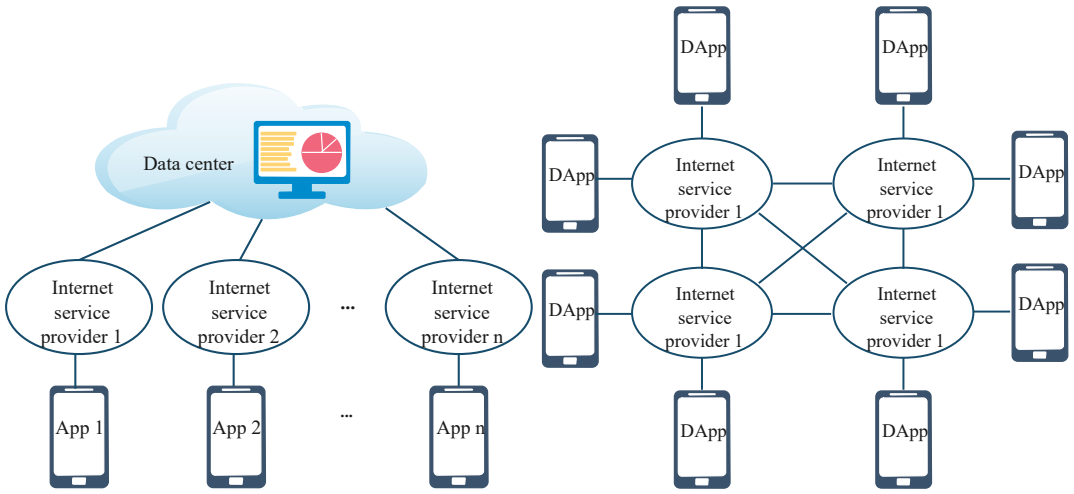
- Applications must be completely open-source with no entity controlling the majority of its tokens.
- The application's data and records of operation must be cryptographically stored in a publicly-

accessible distributed manner. In this way, it can avoid any central points of failure.

- The application must use a cryptographic token - this is required for accessing the application and any contributions should be rewarded with the application's tokens.

- The application will reward contributors in the community according to a proof of value concept which is predefined by standard cryptographic techniques.

Figure 2.9 illustrates the differences of centralized and decentralized application plane. Table 2.2 lists the comparisons between traditional conversational services and blockchain-based conversational services.



(a) Centralized Application (b) Decentralized Application

Figure 2.9: Comparison between application plane

Table 2.2: The comparisons between traditional conversational service and blockchain-based conversational service

Topic	Traditional conversational service	Blockchain-based conversational service
Data plane	Tradition database structure such as table, b-tree, vector, etc.	Chain data structure with cryptographic methods such as hash, asymmetric encryption, etc.
Control plane	Totally or locally centralized control mechanism.	Decentralized control mechanism.
Network plane	Client/Server network through centralized management.	P2P network through distributed recording, transmission and storage.
Application plane	Many large corporate-based Internet entities.	User-centric.

2.4.5 Blockchain-based Architecture in Demand

Based on the above discussion, we can see that contemporary conversational service architectures are showing an inability to efficiently respond to the increasing challenges in many aspects, especially in terms of service security and privacy. We explained the main reasons why blockchain technology can improve the security of traditional centralized-based conversational service.

End-data monopolies. While a data monopoly provides an appropriate business for tech giants, from a user‘s perspective, it is not fair that this data can be freely obtained from end-users and then monetized. (McAfee et al. 2012).

End-surveillance on the Internet. The private data and activity of users is monitored and collected by conversational services, typically without the consent or knowledge of the user. This is at the expense of a user owning and controlling their identity and security.

Permissionless innovation is reintroduced to the Internet. We need to build an open or public application service network instead of private or proprietary services. Then, regardless of where you are and which service or application you use, interoperability and sharing of information is facilitated and transparent.

In summary, the blockchain-based conversation system architecture is to build a decentralized structure with distributed data verification on which modern conversational services can run. The innovation of blockchain-based conversation system architecture is the database technology serving as "the chain of blocks linked using cryptography", which is to provide constant and security connectivity for dynamic network. In addition, the consensus and incentive mechanism of the blockchain will also provide fairness, trustworthy and scalability to upper-layer applications.

2.5 Blockchain-based conversational system

In the previous section, the key concepts of blockchain through different planes in the conversational services were discussed which try to address the issues of security of the information maintained by the network. This section will describe how blockchain technology can be built into a layered conversational service architecture.

2.5.1 Vision of Building Blockchain-based conversation system

This section we presents a vision of building decentralized, multi-tier conversation system for characterizing and standardizing the typical features and main components of blockchain and briefly describes the underlying structure of each plane. As shown in Figure 2.10, blockchain-based conversation system can run on a fully peer-to-peer (P2P) basis. Each node in the network can act as both client and server and compared to current conversational services, clients do not reply on a central server which thereby facilitates interaction. The new architecture is a web of connected nodes which make up the network itself. These nodes communicate with each other to maintain, measure and update the new entries in the database. All nodes work together to guarantee they reach a consensus to provide the network with in-built security.

Data Plane: this plane manages multichain data with related cryptography methods to maintain the blockchain database in an ACID (Atomicity, Consistency, Isolation, Durability) style. The data plane also performs necessary required database actions such as create, insert and update (Siewert 2018). A basic blockchain selects a peer based on the winner of a consensus competition of block hash and it will be authorized to create a new block and add it to the chain structure, encapsulating all transaction data with a specific timestamp generated over the Internet between nodes. For the design of multi-chain databases, the storage structure, data management, verification mechanism and cross-chain anchoring method are four key components. The Merkle tree and block hash are used to secure verification of content in a large dataset, and help to verify the content and consistency of the data while block hash combined with timestamp makes block chain manipulation harder for an adversary. The traceability of the blockchain data is also enabled (Liang et al. 2017). Another aspect in this plane is the anchoring between multichains, with each multichain blockchain having a set of blockchain parameters determining the chain's behaviour. Different blockchains can also use predefined proofs such as Simplified Payment Verification (SPV) (Back et al. 2014) to ensure data security and non-tampered. In this way, data can be transmitted between different blockchains, which engenders more extensive application prospects.

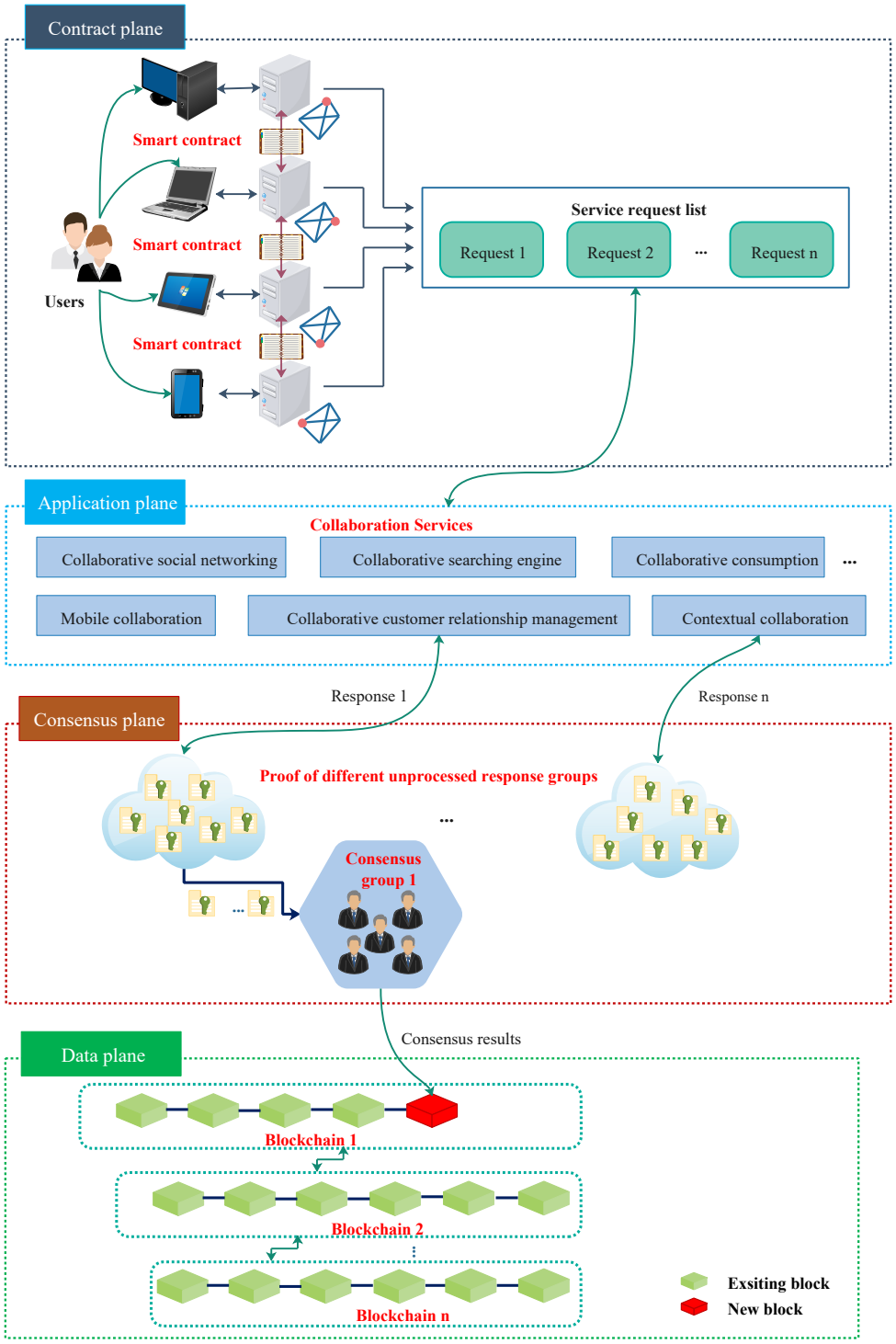


Figure 2.10: The vision of building blockchain-based conversation system architecture.

Based on the decentralized multichain structure, peers are equally privileged without central administrators or hierarchical entities and can be considered as full user-centric and light-weight peers. Any

new datum or block created by one peer will be broadcast to all monitoring nodes in the network. Every node stores all blockchain data, which can be easily synchronized and maintained in the event of the node's failure. In this way, massive amounts of data can be shared amongst completely decentralized Internet entities.

Consensus Plane: this plane packages all consensus algorithms for all Internet peers in the network. These algorithms enable participants to agree on the contents of the blockchain in a distributed and trustless manner. Essentially, a consensus algorithm is used for Internet tasks that can be crowd-sourced. Current consensus algorithms are relatively slow to converge and do not support the satisfactory processing and confirmation speeds required for instant services. Therefore, there is a need to design a reasonable crowd-sourcing mechanism with an incentive capability that enables rewards for each peer across the Internet while ensuring data security (Bozic et al. 2016). Specifically, it is related to intrachain proof and interchain proof, and the overall consensus service is based on the dynamic collaborations between different service providers. Since the transaction verification is the key problems of consensus process, it is better to select verification nodes dynamically rather than the whole nodes. This can greatly increase the cost of malign peers and reduce the communication delay in the consensus process, thus the designing of consensus algorithms could considered the adjustment of workload (such as service transaction volume+ transaction age) to determine the difficulty of mining nodes and the consensus representative selection.

Application Plane: this plane is commonly accessed to provide conversational services. This acts as an interface between users and the underlying planes, where actual applications are defined such as applied data mining, machining learning, intelligent assistants and other Internet applications. Traditional applications follow a centralized client-server model that directly controls the flow of information from a single centre. All individual clients are totally dependent on centralized services such as Google, Facebook, Amazon and other mainstream services (Back et al. 2014) to send requests and receive responses. A decentralized application plane allows different types of applications that use a point-to-point communication model. Designing application is mainly composed with three main modules: a construction module which designs the internetworking mode between multiple service providers to ensure scalable data storage and secure access control, an authority module which is responsible for the permissions related to the contents or contributions of each participants, and a transaction module which is responsible for exchanging the information or value between nodes.

Contract Plane: this plane encapsulates various scripts, algorithms and smart contracts. Users can define self-request, self-verifying, self-executing and self-response rules via a personalized smart contract. The contract plane provides essential services to the decentralized application plane as well as the control

plane, making them programmatic smart properties. For instance, when executing a web service using the HTTP protocol, the contract plane will self-execute and return the corresponding HTTP responses to predefined HTTP requests without any intervention from a third-party. Each response needs to satisfy the consensus algorithms deployed in the consensus plane. After response verification, the new response can be updated in the data plane. The key points to design smart contracts are transaction processing, storage mechanism and complete status identification. The transactions mainly include request and response messages between service providers and users, and when these transactions are transferred into smart contracts, the status identification will be triggered and updated. If the predefined conditions (such as agreed time and event) are satisfied, then smart contracts are executed to guarantee all the chains run the deployed functions automatically.

2.5.2 Technical superiority

The integration of blockchain in the conversation system architecture could solve many problems that the current architectures face. The role of a blockchain-based mechanism for conversation system is detailed by the following aspects:

1) *Improve data security for content storage*

   Many information is very important for each customer during service interaction. Thus, these contents should be clearly identified and data integrity should be ensured. Blockchain can provide a reliable peer-to-peer communication with security and traceable measures over a untrusted network.

2) *Provide a reliable incentive scheme based on consensus mechanism*

   Incentives are what encourage communities of participants to cooperate and create the value that ensure the success of conversational services. Another advantage of this integration is the possibility to make incentive trusted decisions since the blockchain can ensure that all participants of a decentralized network share identical contents and get consensus. This assurance can allow the system to reach an agreement over the whole network and to have global collaboration between the different entities.

3) *Provide effective corporation to support multiple conversational services*

   Sharing of multiple conversational services is related to multiple corporations, and the exchange of value will also involve multiple accounts. Blockchain can support complex transactions by simply using smart contracts which have excellent scriptable programmability, and this would increase the fundamental baseline extensibility needed to support different types of conversational services.

2.6 Key Blockchain-based Conversation system Requirements

As stated in previous section, a layered security conversation system architecture can be built through blockchain technology. However, from a research viewpoint, there are several key technical requirements that need to be addressed for blockchain-based conversation system to reach their full potential. The key requirements are explored and summarized in the building of blockchain-based conversation system architecture as shown in Figure 2.11.

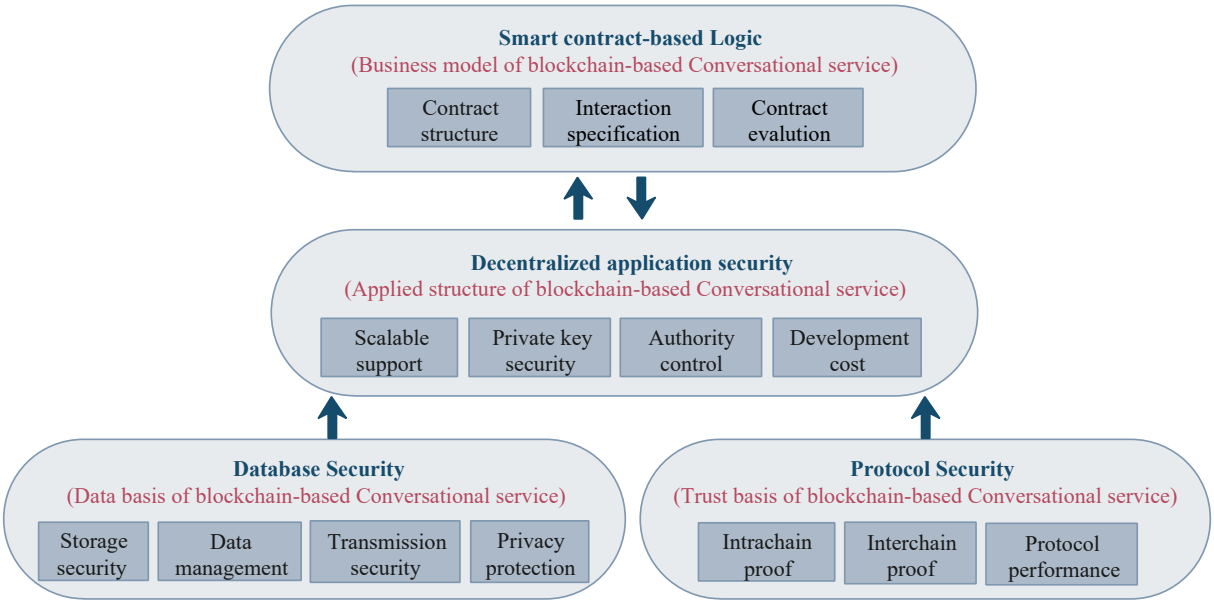


Figure 2.11: Key technical requirements in Blockchain-based conversational service.

2.6.1 Database Security requirements

The blockchain database has shown a proven robustness to data security and integrity in cryptocurrencies, which not only supports a single blockchain, but also provides sidechains as well as multichains used by all participants through secured cryptographic protocols (Bozic et al. 2016, Raval 2016, Greenspan 2015). The advent of decentralized databases built on blockchain technology creates new requirements, as they will exchange massive volumes of data that need to be stored and managed. The following requirements are investigated:

Storage security: decentralized storage needs to meet the demands of storing high volumes of data across the Internet. Blockchain's linked storage structure allows for one chain on the whole network. All coincident transactions are kept in the same block based on a consensus algorithm, and in the case of Bitcoin, a block is created every few minutes. However with the exponential increase of technology

usage, from the point of technical implementation, there are three main methods: sidechains, sharding and Directed Acyclic Graph (DAG). Rootstock (Lerner 2015), Alpha (Pal 2017) and Liquid (Zhou et al. 2021) are typical examples of using sidechains, which allow tokens and other digital assets from one blockchain to be securely used in another separate blockchain and then be moved back to the original blockchain if needed. Zilliqa (Meneghetti et al. 2019), Rchain (Eykholt et al. 2017) and Quarkchain (Aldakheel et al. 2019) use the sharding mechanism to scale up, which divides the super blockchain network into several sub-chain networks (each sub-chain network we call a shard) consisting of part of peers. IoT Chain (ITC) (Xie Zhuopeng n.d.), Byteball (Popov n.d.*a*) and IOTA (Popov n.d.*b*) are the most applicable examples of the DAG structure.

These new implementations are scalable, light-weight and decentralized, making them more suitable for large-scale networks and they also support different types of transactions being recorded on different chains simultaneously. The storage potential of enhancing the single-chain blockchain storage into sidechains, sharding and DAG (Popov n.d.*b*) structures can be seen in Figure 2.12.

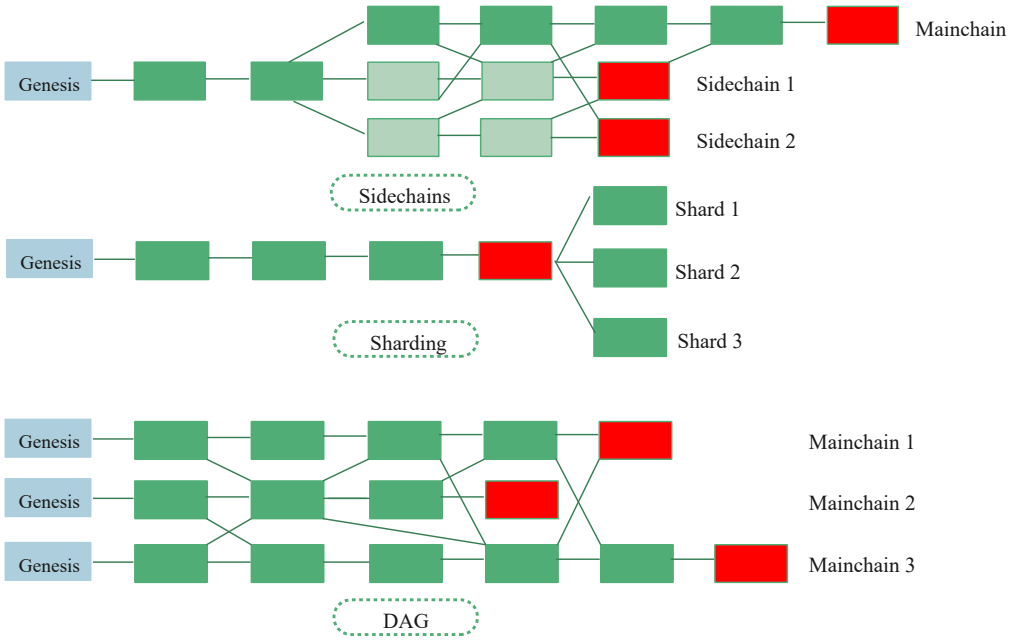


Figure 2.12: Storage structure between sidechains, sharding and DAG.

Data management: in a traditional database, a client can perform four basic functions on data: Create, Read, Update, and Delete (CRUD commands). Since blockchain data is permanently stored and is immutable, the operations associated with blockchain are creating and reading, which means that there is no native deletion or update. Reading can query and retrieve existing data from blocks indexed by their hash value with some other attributes. Writing is delayed by waiting for block creation, and an additional

mechanism is required to implement the concept of deletion and update (for example, flagging transactions as stale). A public blockchain database is a read-uncontrolled as well as write-uncontrolled database, which means any client can read a block in the existing chains, and write a new block into the chains (subject to consensus) (Shahin et al. 2018). However, with the existing technology, write operations are slow due to the transaction confirmation mechanisms which take several minutes to complete. Therefore, faster and more intelligent methods are required to maintain data with blockchain-based conversational service databases.

Transmission security: blockchain databases use advanced cryptographic techniques to ensure data transmission security. It involves at least two levels of security protection. Firstly, the global states are protected by a Merkel tree where the root hash is stored in the block header. Furthermore, the block history is also protected through a chain of cryptographic hash pointers (Dinh et al. 2018, CHEN et al. 2017). Hashing is also used in encryption of transactions. There are several typical cryptographic algorithms, such as MD5, SHA1/SHA2 and SM3 (Dinh et al. 2018). As indicated by Table 2.3, hash functions like MD5 and SHA1 are officially insecure, and SHA2 and SHA3 are the most popular hash functions used in blockchain databases. The Merkel tree helps achieve rapid and secured transaction verification, while the hashing and time-stamp enable integrity and traceability during transmission between peers in the network.

Table 2.3: Key features of Typical Hash functions used in blockchain database

Algorithm	security	Arithmetic speed	Output length
MD5	Low	Fast	128
SHA1	Medium	Medium	160
SHA2	High	Lower than SHA1	256
SHA3	High	Lower than SHA1	256

Privacy protection: as previously discussed, blockchain-based data storage and transmission is transparent, and users can use a digital signature to protect their privacy through the use of a public and private key pair. Public keys can be shared with everyone while private keys are kept secret. Either of the keys can be used for message encryption while the other key is used for message decryption. RSA (Rivest Shamir Adleman), ECC (Elliptic-curve cryptography) and SM2 (SuperMemo) are among the most common asymmetric encryption methods used in blockchain systems (Zyskind et al. 2015). Table 2.4 lists the key features of these digital signature methods. These asymmetric encryption methods should be further strengthened in a huge number of conversational services with multichain interaction.

Table 2.4: Key features of typical digital signature methods used in blockchain database

Algorithm	security	Maturity	Arithmetic speed	Resource consumption
RSA	Low	High	Slow	High
ECC	High	High	Medium	Medium
SM2	High	High	Medium	Medium

2.6.2 Protocol Design requirements

Since a blockchain database supports both single chain and multichain structures, there is a need to design and apply different protocols to ensure trust is inherent. The following requirements are discussed for consensus protocols used for intrachain and interchain communication.

Intrachain proof protocol: The consensus protocol for single blockchains is used to achieve agreement on a single data value among distributed processes or systems. The most common consensus protocols used for single blockchains include: PoW, PoS, DPoS, Paxos, PBFT and DBFT.

- PoW: Proof of Work is one of the first utilized consensus protocols that is computationally based, requiring miners to find the solution to a puzzle. Several cryptocurrencies utilize a variant of this protocol (Nakamoto n.d., King 2013, Duong et al. 2018). It is a data item that is time-consuming to produce but easy to verify by others which satisfies specific necessities (Alla et al. 2018, Gervais et al. 2016).

- PoS: Proof of Stake is a proposal that determines who will add the next block into the blockchain based on how much stake a miner has in the network (King & Nadal 2012) - in other words, mining is done by stakeholders in the ecosystem who have the resilient motivations to be decent stewards of the system (Vukolić 2015, Kiayias et al. 2017).

- DPos: Delegated Proof of Stake is a newer consensus structure where users select some delegate nodes which confirm the validity of a block (Larimer 2014). The network performance, resource consumption and fault tolerance of DPoS are similar with PoS (Larimer 2013).

- Paxos: Paxos is a consensus protocol based on a leader role. A leader node has absolute authority and it allows other nodes to participate in supervision. All the nodes in the network have a general access mechanism. However, during the process of selection, malignant nodes cannot be allowed. Hence, fault tolerance is not available in Paxos (Hunt et al. 2010, Birman et al. 2010).

- PBFT: similar to Paxos, Practical Byzantine Fault Tolerance (PBFT) uses permissive voting, with the principle that the minority is subordinate to the majority (Sankar et al. 2017). In contrast, the

consensus algorithm allows a 33.3% fault tolerance (Madigan et al. 2012).

- DBFT: Delegated Byzantine Fault Tolerance (DBFT) is similar to PBFT where the main difference is based on including a leader driver with delegation to improve the efficiency of data processing (Jeon et al. 2018).

Interchain proof protocol: an efficient and secure communication protocol over the Internet and is the most in-demand technology for blockchain-based conversational services. As multichain blockchains can allow for large storage capacities, together with higher data integrity and transparency, multichain is a suitable solution (Cachin & Vukolić 2017). Among multichain technologies, cross-chain communication is one of the key issues. Key types of cross-chain technologies are outlined below.

- Notary schemes: these are the most common schemes used for routing payments across diverse digital ledgers through the separation of receivers and senders from the risk of intermediary failures. This protocol is invoked by hosts over higher-level protocol modules in an interledger environment (Thomas & Schwartz 2015). Figure 2.13 shows the basic layered structure for a notary scheme.

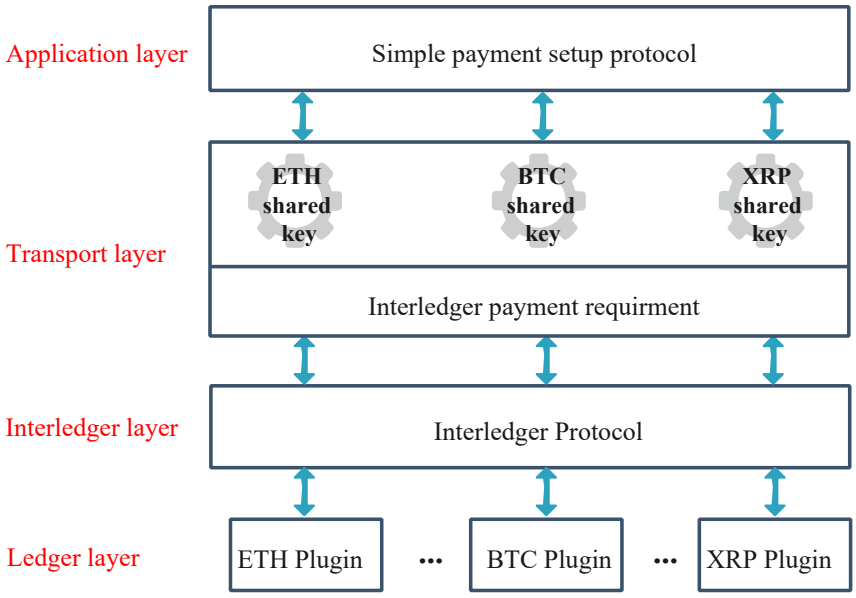


Figure 2.13: The notary scheme layered structure.

Referring to the Figure 2.13, connectors act as a notary to build the communication between interledgers deployed in the different blockchain platforms. For instance, in Ripple, the Notary module would call on a local ledger module which would create a Ripple transaction with the interledger packet attached to the Ripple Consensus algorithm (Lee 2016). Then, the Ripple address would be derived from the interledger address that might be connected to other ledgers via

the local ledger interface.

- Relays: this technology uses building blocks that allow contracts to securely verify blockchain transactions without any intermediaries. They can also act as a smart contract that stores block headers. Then, these block headers are being used to build a mini-version of the blockchain (Wang, Zhou, Wang & Finestone 2018) (refer to figure 2.14). Bitcoin also uses this method to achieve Simplified Payment Verification (SPV) light wallets. The work flow is divided into three steps:

  i) relayers constantly submit blockchain headers;

  ii) transactions are submitted to be verified;

  iii) verified transactions will be replayed to the smart contract.

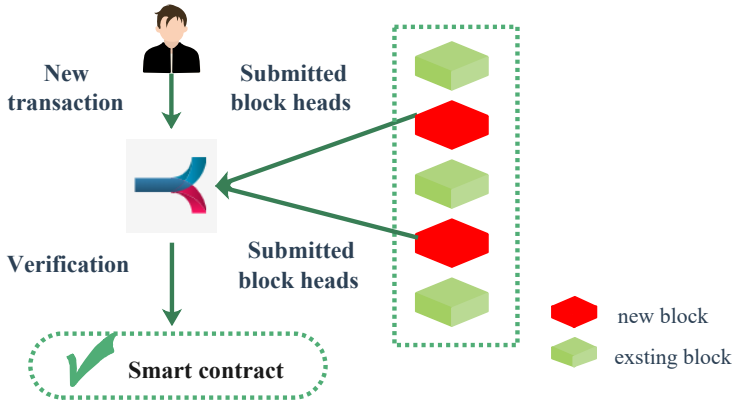


Figure 2.14: Relays design and construction.

Relays belong to the early stage of cross-chain communication technology. They combine two different blockchains with a defined smart contract. The applied trust model is similar to the single blockchain and chains do not fail or suffer from 51% consensus attacks. A typical example implemented by relays is BTC (Bitcoin) relays that connects Ethereum and Bitcoin using a smart contract (Chow 2016), where clients can pay for Ethereum usage via Bitcoin payment. Another example is RootStock (RSK) (Lerner 2015) which is a smart-contract platform that incorporates a Turing Complete Virtual Machine (TCVM) with Bitcoin. Relays also provide some network enhancements such as better scalability and faster transaction features which will also enable new usage scenarios.

- Hash-locking: this is a key technology of the lightning network. Single blockchain has limitations such as the transaction rate (of the order of a few transactions per second in the whole network), and the verification of new blocks which require relatively long time durations by consensus nodes (Poon & Dryja 2016, Deng et al. 2018). Both these problems bring difficulties when

extending the application capabilities of blockchain-based conversational services. Hash locking provides an extended channel that restricts the spending of an output until a specified piece of data is publicly revealed. Hash-locking has the useful property that once any hash-lock is opened publicly, any other hash-lock secured using the same key can also be opened (Buterin 2016) (Figure 2.15).

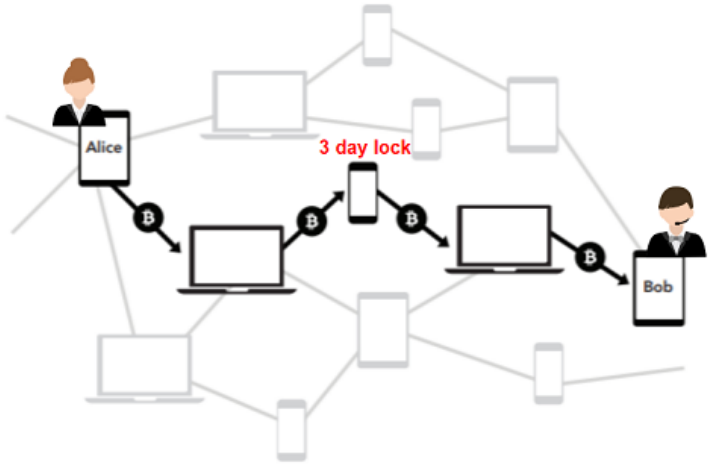


Figure 2.15: Hash-locking transaction example.

For instance, if two users (Alice and Bob) make a Hashed Timelock Contract (HTLC) protocol before communication, the blockchain system will lock the lightning network between them (Alice and Bob), until Bob can return a hash value within 3 days. If this hash value is correct, Alice can transfer money to Bob immediately. Therefore, if two peers pre-set a hash-lock contract, then they can achieve instant and multiple transactions between each other. However, although hash-locking can realize the exchange of digital assets, it cannot support cross-chain contracts. Hence, hash-locking applications are limited.

- Distributed private key control: this is a hybrid protocol that combines some single blockchain protocols together. Private assets can be mapped to a public blockchain through a distributed private key and control technology, which can realize lock-in and unlocked modes. A lock-in model is the process focusing on retaining the control and mapping of assets, while unlocked is the reverse operation of the lock-in process, allowing control power to be returned to the owner. Figure 2.16 shows the basic function of distributed private key control (Deng et al. 2018).

  As an example, fusion is a popular distributed private key control platform. Fusion ensures that nobody can access the complete private key, making sure that no single node can obtain the control of the completely digital ledger. In addition, Fusion is based on the Hierarchical Hybrid Consensus Mechanism (HHCM) combining the PoS and PoW blockchain protocols, and it utilizes parallel computing to group nodes, thereby achieving a favourable balance of efficiency and safety.

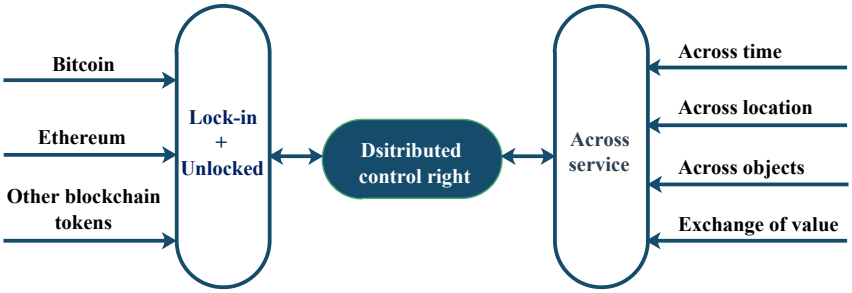


Figure 2.16: Basic structure of distributed private key control.

- Notary schemes + Relays: this key type combines both technologies. Relays are first used to build an efficient communication channel and Notary schemes aim to achieve instant transactions between peers in the network. One typical instance is Ether Universe (Meshcheryakov et al. 2020) (as shown in Figure 2.17). Ether Universe connects different blockchain networks such as Ethereum, Bitcoin, EOS and others via 'connectors' used in Notary schemes and 'verification' used in Relays.

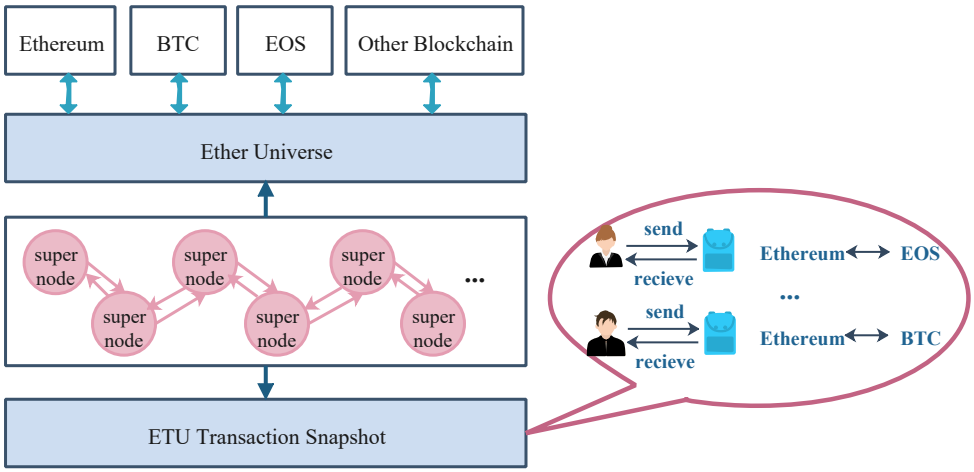


Figure 2.17: Basic structure of Ether Universe combined with notary schemes and relays.

Ether Universe inherits the advantages of EOS, which can process millions of transactions per second and generate corresponding transaction snapshots at the same time. Ether Universe is a very recent addition to the cross-chain platform which requires further evaluation.

Protocol performance: Consensus algorithms are designed to establish reliability in a network involving multiple unreliable nodes. For the consensus algorithms used in single blockchain, the protocol performance is mainly analyzed based on the average confirmed efficiency, resource consumption and tolerance power (Yang et al. 2018) presented in Table 2.5. From Table 2.5, it can be seen that PoW, PoS and other consensus algorithms are inefficient with associated issues of serious energy consumption. Hence, these algorithms cannot meet the performance requirements of blockchain-based conversational services.

Table 2.5: Comparison with different single-chain consensus protocol.

Consensus	PoW	PoS	DPoS	Paxos	PBFT	DBFT
Year	2008	2012	2014	2015	2015	2016
Average confirmation time	about 10 minutes	about 60 seconds	about 3 seconds	about 1 second	about 1 second	about 1 second
CPU usage	High	Medium	Medium	Low	Low	Low
Tolerance power	$\leq 25\%$	$\leq 51\%$	$\leq 51\%$	$\leq 51\%$	$\leq 33.3\%$	$\leq 33.3\%$

Considering the poor consensus and energy performance of most current intrachain protocols, the design of new intrachain protocol should be satisfied with the following requirements:

1) Dynamic verification: is able to perceive and adjust the mining structure for different networking environments. In addition, dynamic verification also reflects the more efficient usage of computing resources such as CPU load, memory, bandwidth and so on. Hence, the performance of an intrachain protocol should have a stable longer-term decrease use of resources.

2) High-throughput and low delay: high-throughput means the intrachain protocol can process more verification requests per unit time and the low delay is related to the transaction cost. The intrachain protocol should optimize the user experience and reduce waiting times.

3) Low power consumption: to support large-scale conversational services, the design of node selection strategy, grouping verification and node management can be used to reduce power consumption.

Here, the key features between the different cross-chain communication technologies in Table 2.6 are compared and discussed by the following presented criteria.

- **Trust model:** proof principles used between separated chains.
- **Usable for cross contract:** the difficulty level of smart contracts deployed into multichain structure.
- **Transaction speed:** the transaction processing performance during mining.

Considering the above criteria for an interchain protocol, Notary schemes and Hash-locking have difficulty in support cross-chain smart contracts, and thus, they have poor scalability. Relays have low transaction speed and high delay, which is also not suitable for various conversational services. It is necessary to design a hybrid interchain protocol to support concurrent processing of diversified services that satisfy the following requirements.

1) Anchoring between multi-chains to guarantee non-tampering: the transactions between chains should be linked by two-way peg[1] or other similar strategies to ensure reliable transmission and

[1]two-way peg enables interchangeability of digital assets at a predetermined rate between the two chains.

Table 2.6: Comparison with different cross-chain consensus protocol.

Types	Notary schemes	Relays	Hash-locking	Distributed private key control	Notary schemes + Relays
Typical Applications	Interledger	BTC, RootStock, Polkadot	HTLC	Fusion	Ether Universe
Trust model	Majority of notaries honest	Chains do not fail or get"51% attacked"	Chains do not fail or get "51% attacked"	Hybrid consensus protocol	Majority of notaries honest + Chains do not fail or get "51% attacked"
Usable for cross-contract	Difficult	Easy	Difficult	Easy	Easy
Transaction speed	Low	Low	High	High	High
Popularity	Launched in 2012, well-known	Launched in 2015, well-known	Launched in 2016, not well-known	Launched in 2017, not well-known	Launched in 2018

avoid double cost.

2) Efficient verification of cross-chain transactions: a shorter block interval can make transaction verification more efficient, but it may cause increased chain forking that reduces the network availability. Thus, the design of an interchain protocol should consider the trade-off between verification time and the number of forks.

3) Cooperative consensus based on dynamic construction strategy: the consensus nodes selected from different chains are used to build a set of verification nodes. The dynamic construction strategy should be based on the computing power, the credibility of the node and other factors to make sure that the selection of verification nodes is uncontrollable.

2.6.3 Application requirements

In this subsection, several key requirements of applications for various conversational services are listed.

Scalable (massive) user support: At present, basic conversational services such as web-based shopping sites like Amazon and Internet email hosts like Gmail have a massive number of user accounts. Therefore, in order to deploy a new conversational service architecture on the basis of the blockchain platform, the architecture has to support massive numbers of users, and avoid the resulting problems related to network performance, while also giving consideration to expandability storage.

Security of private keys: the user experience is an important indicator of conversational services. It is inefficient and possibly insecure to use a haphazard, guessable string as an account or password identifier for each user. In addition, if a user loses their authentication details, there is a need for authentication mechanisms to re-establish the identity. Contemporary systems apply two-factor authentication. However, it is desirable to have a set of security mechanisms to store private keys combining the blockchain platform and application layer (Zyskind et al. 2015).

Authority control: data sharing and transparency are very sensitive topics for business services. Simultaneously, as a mutual trust between peers is being built, there is a need to guarantee the privacy of commercially-sensitive information as well as individual privacy, as they are included in the basic philosophy of the blockchain-based service architecture.

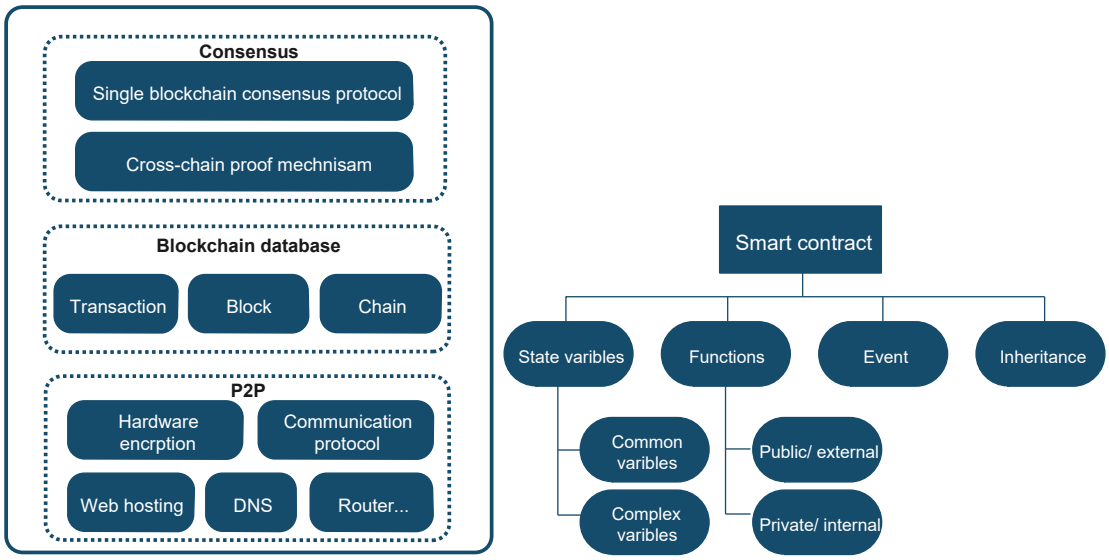
Development cost: the convenience and reliability of application development determines the success of blockchain deployment. During the development phase, there is a need to put into consideration the costs of development including the technical, time and labour costs (Zmaznev et al. 2018).

2.6.4 Contract requirements

A blockchain-based conversational service architecture provides two levels of contracts: standard and smart contracts. A standard contract is suitable for simple scenarios and is always deployed or encapsulated when the blockchain is initially created (only simple commands are supported). A smart contract aims to solve more complex scenarios and it exposes many API interfaces for arbitrary programming that developers can use to make complex agreements between different nodes (Christidis & Devetsikiotis 2016). The key requirements of standard and smart contracts have been outlined in Figure 2.18.

Contract structure: the standard contract can be considered as the cryptography mechanism used inside the blockchain platform as described in previous sections. The standard contract cannot be updated and deleted after being deployed in the blockchain system. On the other hand, the smart contract includes fully-featured scripted programming, made up of a set of rules running on top of a blockchain-based system. The smart contract is also proposed to reduce transactional costs and guarantee a greater degree of security (Christidis & Devetsikiotis 2016). The main structures used in standard and smart contracts are shown in Figure 2.18.

Interface specifications: the contract interface should be designed according to the blockchain database model. Operations related to contracts can be classified into two levels: static and dynamic. The



(a) Standard contract in blockchain (b) Smart contract in blockchain

Figure 2.18: Basic components of standard contract and smart contract.

static level aims to define the relationship between users and objects. For instance, a standard contract can be used to create an account and declare the owned assets. The dynamic level focuses on operations between users and users or users and objects. For instance, a smart contract can be used to define restrictions with regard to asset transfers, updating account information, access control and so on.

Contract evaluation: although smart contracts are used in many blockchain platforms and are driven by many different types of services, it is necessary to determine the evaluation measures of smart contracts (Idelberger et al. 2016). After understanding the ways to apply smart contracts with detailed insights, there is a need to measure the performances and challenges when they are deployed, such as formal descriptions, contract model verification, consistency tests and so on (Bhargavan et al. 2016, Reza M. Parizi 2018) (Figure 2.19).

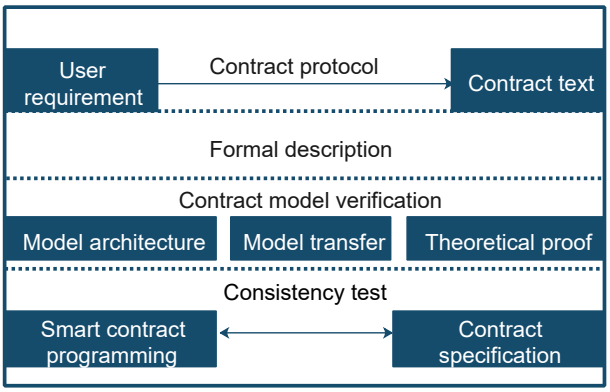


Figure 2.19: Contract evaluation requirements

2.6.5 Blockchain Trends for Future Conversation system

Over the past few years, along with the rapid development of the Internet, there are five main Internet technologies which have mainly influenced the early development of blockchain (Peters et al. 2015) (shown in figure 2.20), including TCP/IP (Cerf et al. 1974), Routers (Burstein & Pelavin 1984), Web applications (Shuchun et al. 2000), P2P (Barkai 2000) and information security technology (Halevi & Krawczyk 2006). Based on these five main influences, blockchain attempts to build a decentralized structure to achieve many applications using cryptographic methods.

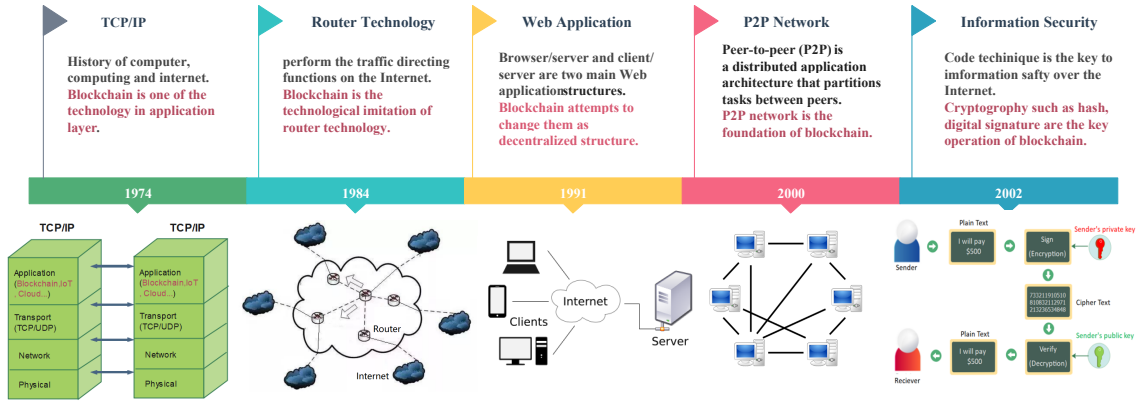


Figure 2.20: 'Development of Internet' VS 'Development of Blockchain"'.

It can be seen that the TCP/IP protocol is the basic technology and de facto standard for the transport and networks layers of a layer-based approach for internetworking, but now blockchain technology is one of the new technologies in the associated application layer. Blockchain is the technological imitation of router technology from the network layer to the application layer, that performs traffic routing decision functions required on the Internet. With the development of web applications, two main application structures have emerged: browser/server and client/server model. However, both models are based on centralized or locally centralized controllers to concentrate on conversational service or applications, while blockchain attempts to change them to a decentralized structure. In 2000, a P2P network was proposed to partition tasks or workloads among peers which is the foundation of blockchain technology. Then, following the many information security technologies used for conversation system, blockchain used several cryptographic methods to build a transparency and trustworthy mechanism to support transactions between different peers. Thus there is an inextricable connection between the development of the Internet and blockchain technology.

2.6.6 Trends in Blockchain- Systems

Up to now, blockchain technology has been steadily developing from the original Bitcoin protocol for digital currency to the second generation Ethereum platform integrated with smart contracts (Swan 2015). Today, we are in the process of building what is unofficially termed blockchain 3.0 and future-generational blockchain 4.0 (Zhao et al. 2016, Joshi 2017). In this section, we provide a simple description about how the technology is evolving from its initial form, to become a fully-edged globally distributed system as shown in Figure 2.21.

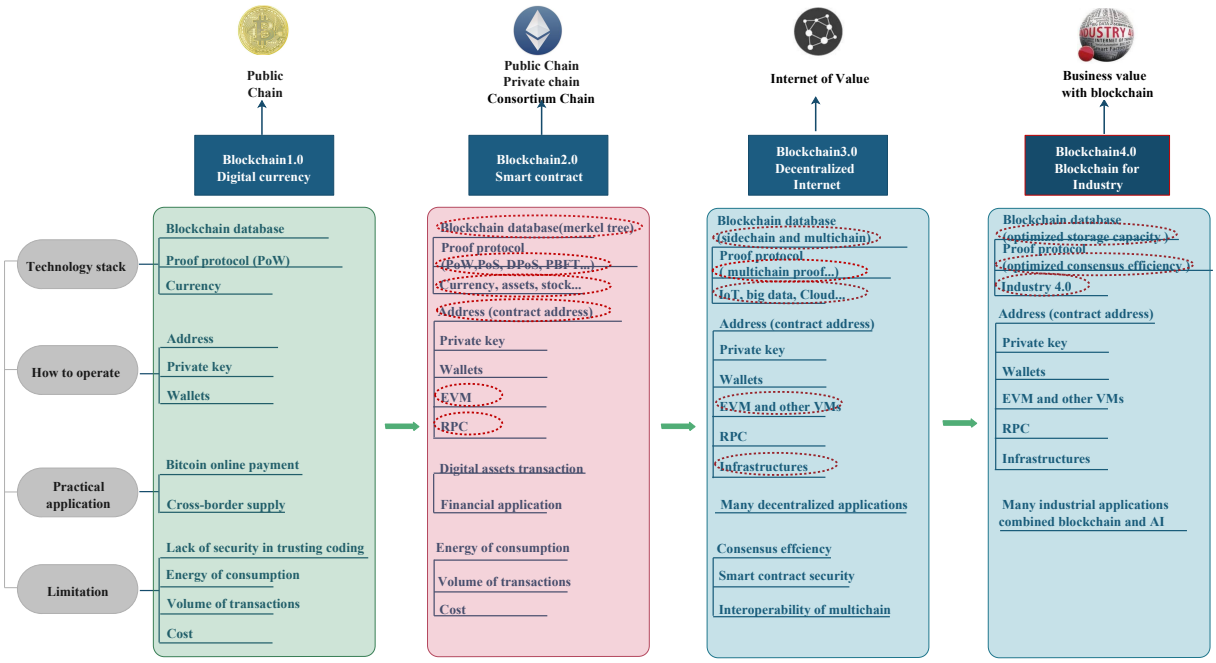


Figure 2.21: The evolution of Blockchain system.

Blockchain 1.0 is completely dedicated to the digital currency. The typical platforms that are supported are the mining of Bitcoin and other crypto-currencies such as Litecoin (Reed 2017), Dogecoin (Young 2018) and so on. The consensus algorithm utilizes Proof of Work (PoW) which is only used in the public chain. Blockchain 1.0 guarantees distributed storage, allows data sharing between nodes, and enables transparency in transactional processing (Yang et al. 2018).

In Blockchain 2.0, some new cryptographic methods such as the Merkle tree were added into the data plane to more efficiently manage the transactions. In addition, apart from PoW and PoS, other consensus algorithms used in public chains, private chains or consortium chains, such as DPos and PBFT, were proposed to reduce the volume of transactions (Zheng, Xie, Dai, Chen & Wang 2017, Larmuseau & Shila 2019). The most important improvement was the utilization of the smart contract, which automatically executes small computer programs when certain conditions are met (Yang et al. 2018). Smart contracts

aimed to reduce the cost of verification and execution, while aiding fraud prevention. The most prominent system in this version of blockchain was Ethereum, proposed in 2013. This version allows the formation and transfer of digital assets and other financial applications. The main limitations of Blockchain 1.0 and 2.0 are the energy consumption,volume of transactions and cost (Yang et al. 2018).

In order to tackle the limitations in blockchain 1.0 and 2.0, a third generation of blockchain platforms was proposed to support different blockchain data structures, proof protocols and the development of various areas rather than financial applications. However, it still has some limitations such as the efficiency of consensus, security of smart contract and interoperability of multichain (Zheng et al. 2016, Lin & Liao 2017).

With the rise of new industrial technology, known as Industry 4.0, a fourth generation of blockchain platforms is being presented to provide ideal solutions to satisfy business demands. Blockchain 4.0 aims to improve the consensus efficiency, the scalability of blockchain networks, the energy requirements of computation and so on, thereby tailoring blockchain to real, contemporary and future environments.

2.6.7 Challenges and Future

Based on the above discussions, the evolving key requirements which enable blockchain to be able to communicate and interoperate over the conversation system, maintain a global and reliable repository of information (Zheng et al. 2016) can be found. However, blockchain is also faced with multiple challenges and research problems that need to be resolved. Generally, three criteria are always used to assess the blockchain technology: decentralization, scalability and consistency. There is a tradeoff among these three characteristics, for example, the applications based on Bitcoin and Ethereum platforms are decentralized and consistent, where every full node stores all the data without centralized control, but they suffer in the lack of true scalability (which is exhibited by the duration of several minutes needed for one block confirmation). To apply the blockchain-based internet service, we summarize future challenges mapped to proposed key requirements as shown in Table 2.7.

From Table 2.7, it can be seen that there are a few challenges that need to be addressed before the current blockchain technologies can concurrently assure decentralization, scalability and consistency with billions of transactions in each second. Here, we outline the main challenges to six areas:

1) Storage capacity: in blockchain, there is a requirement for all transactions to be stored in each node and this record is immutable, ensuring data integrity and continuity. However, this introduces the problem of excessive system storage due to the characteristics of non-erasable and distributed

Table 2.7: Key requirements of blockchain-based conversational services mapped to existing blockchain elements

Type	Key requirements	Exsiting blockchain elements						Typical examples
		Data structure	Consensus protocol	Multichain proof protocol	Decentra -lized protocol	Standard contract	Smart contract	
Database	Data security	✓	✓	✓	✓	✓	✓	(Pinyaphat 2018) (Namasudra et al. 2021)
	Data storage capacity	×	×	N/A	N/A	N/A	N/A	
	Data management	✓	✓	✓	✓	✓	✓	
Protocol	Confirmation time	N/A	✓	✓	N/A	✓	✓	(Velliangiri 2020) (Choo et al. 2020) (Azbeg et al. 2021)
	Performance and efficiency	N/A	×	×	N/A	N/A	N/A	
	Resource consumption	N/A	×	×	N/A	N/A	N/A	
	Tolerance power	N/A	✓	✓	N/A	N/A	N/A	
Application	Scalbility	×	×	×	N/A	N/A	N/A	(Mohanta et al. 2019) (Sharma et al. 2021) (Giordano 2021)
	Privacy and security	✓	✓	✓	✓	✓	✓	
	Data management	×	×	×	×	N/A	✓	
Contract	Transaction fee	N/A	N/A	N/A	N/A	N/A	✓	(Yang et al. 2018) (Khan et al. 2021) (Vacca et al. 2021)
	Programming performance	N/A	N/A	N/A	N/A	×	×	

storage. Therefore, there is a need to design and develop an optimized model of decentralized but robust, reliable and load-balanced storage to allocate data based on the performance of individual nodes.

2) Consensus performance and efficiency: the consensus protocol plays a key role in the scalability of blockchain networks. However, the current consensus methods always require long verification times for transactions, even when there is a relatively small number of nodes. It can be seen that the performance and efficiency of current consensus protocols needs to be improved.

3) Protocol scalability: current blockchain protocols are effective in securing and managing the data stored within the network. However, newer systems fail to scale after some threshold of record and network size (Yang et al. 2018). In order to maintain a coherent and synchronized state of information, a blockchain data structure, in particular for multichain data should be provided to enable communication in a secure and efficient manner without affecting security. This also involves the challenge of both identifying and determining the number of nodes that should have a transaction validation role in order to ensure the best protocol efficiency.

4) Resource consumption: since a small fee is required as an incentive to pay miners for maintaining the distributed ledger (by solving a computationally-expensive problem), this scheme is not satisfactory for massive volumes of transactions due to the prohibitive power (and fiscal) cost. As a consequence, there is a need to seek diminished global power consumption.

5) Personalization mining: providing methods to personalize blockchain for conversation system with multiple conversational services is another important challenge. Artificial intelligence (AI) algorithms can help to solve this by making different parts of the blockchain 'smarter'. For example, node behaviour can be learned via their history of actions to make intelligent decisions. In another example, deciding whether a node should be used in transaction verification or determining the weighting/contribution level of different nodes in the whole network is challenging.

6) Contract performance: the contracts used in blockchain-based systems are computer programs intended to facilitate, verify, or enforce the negotiation or performance of a prior agreement. Unfortunately, current smart contracts do not use the full potential of arbitrary programs, which would allow for a much more semantically-rich environment and a lack of associated contract evaluation. When an arbitrary contract code is enabled, the code requires a rigorous and robust compilation and evaluation system to determine contract pre and post-conditions. Otherwise, the fulfillment of a contract may be vague and subject to unwanted side effects or errors. Another limitation is that contracts cannot change what should in essence be stored due to the current immutability of blocks (or the underlying immutable database metaphor). A layer enabling mutable objects to be stored (distributed and decentralized) is also required but not at the expense of the trust and integrity of the data.

2.7 Summary

In this Chapter, we have conducted a comprehensive survey on current conversational service architectures together with blockchain technologies used to understand the challenges of the conversational service architectures and the benefits of blockchain compared with traditional centralized-based mechanism. We presented the vision of building a blockchain-based conversation system which was designed to achieve trustworthy conversational service in a decentralized manner, then discussed its key technical requirements from different aspects related to the proposed architecture, and analyzed the trends and challenges mapping to these key requirements.

The main purpose of this survey is to guide more detailed and innovative solutions to implement the future conversation system. This style of decentralized conversational service will not only meet the massive information requirements of contemporary and emerging systems, but also coupled with the secure, fair and effective environments such conversation systems are currently lacking.

Chapter 3. A decision model for blockchain applicability into conversation system

Conversation systems usually suffers from the challenge of knowledge management by multiple human experts. As discussed in Chapter 2, the current mechanism used in conversation systems is always based on centralized conversational services. This may be problematic considering transparency and security. Blockchain solutions are currently being proposed to improve security and efficiency in different domains. However, there are various blockchain platforms with different characteristics, and conversation systems implemented using blockchain platform are not yet in place. In this section, we clearly identify the requirement analysis of the conversation system and present a decision model for identifying the best fitting blockchain platform for conversation systems.

3.1 Introduction

The ability to query Knowledge Base (KB) is essential in conversation systems and the KB interpretation requires human inputs. To ensure the quality of knowledge base, it is better to use multiple experts to share the individual knowledge together (Pei et al. 2019). However, the current mechanism of knowledge management from multiple human experts in the conversation system is based on centralized servers, and the main challenges of the centralized mechanism include knowledge sharing and contribution assignment (Keary 2012).

- Knowledge sharing: Traditional knowledge exchange methods are always based on the central server or a third party to collect and transfer knowledge between experts. But this strategy can only support open-source information and for many conventional scenarios, such as medical training, psychological consultation, and travel services, etc., they may include the user's identity information and privacy task solutions with different authorities and permissions. Therefore, it is difficult to manage multiple experts and create knowledge acquisition in a trusted and secure manner by using

a centralized sharing mechanism.

- Contribution assignment: The contributions for each expert, such as the number of successful transactions, and the average customer interaction time of added scenarios, should be stored in a trustworthy and secure manner. The information in traditional mechanisms is always stored in log files that can provide an audit trail but are easily erased or alterable without a trace. Thus, centralized control is not ideal for contribution assignment.

Contrary to traditional centralized mechanisms, blockchain technology is based on decentralized transaction and data management, which can provide anonymity, safety, and data integrity (Wurster et al. 2018, Bach et al. 2018). Blockchain combines with multiple technologies to ensure an immutable, irrevocable, and traceable blockchain ledger (Viriyasitavat & Hoonsopon 2018, Sáez 2020), making this field a vast research area to deal with the limitations and enhancements in current knowledge-based conversation system (ur Rehman et al. 2020). Once established a blockchain-based solution is the right underlying technology, we face a significant challenge of selecting one suitable blockchain platform available for the conversation system. The selection process of blockchain has to consider and fit to the demands and problems of knowledge-based conversation system. In addition, the number of blockchain platforms keeps increasing and the features of blockchain platform also keep improving, therefore, the blockchain platform selection must be adapted keeping collecting and updating. If there are some new blockchain platforms in the market, the knowledge regarding new blockchain platform can be quickly organized and evaluated into selection process when it needs to be applied into conversation system. At present, there are no feasible decision model to support the selection of the most suitable blockchain platforms when applied into knowledge-based conversation system.

3.2 Related work

In this section, we mainly review some state-of-art works in two primary areas: evaluating and selecting for blockchain platforms and decision-making methods.

3.2.1 Evaluating and selecting of blockchain platforms

Several studies point out evaluation and comparison of different blockchain platforms (Dinh et al. 2017, Hileman & Rauchs 2017, Maple & Jackson 2018, Kuo et al. 2019, Farshidi et al. 2020). Dinh et al. (Dinh et al. 2017) proposed a benchmarking framework for evaluating private blockchain systems, which

contains workloads for measuring the data processing performance, and workloads for understanding the performance at different layers of the blockchain. Hileman and Rauchs (Hileman & Rauchs 2017) presented a global benchmarking study to provide better understand current areas of focus, attitudes toward the blockchain technology and challenges that need to be answered. Maple and Jackson (Maple & Jackson 2018) focused on the assessment of different types of blockchain to provide guidance to choose best blockchain solution, which includes analyzing blockchain essential technical features, outlining blockchain building blocks and comparing multiple blockchain platforms. Kuo et al. (Kuo et al. 2019) introduced a comparison of popular blockchain platforms using a systematic review method, and provided a reference for selection of a suitable blockchain platform given requirements and technical features that are common in healthcare and biomedical research applications. Siamak et al. (Farshidi et al. 2020) designed a decision support method for blockchain platform selection and used three industry case including ShareCompany BIQH, DUO and Veris Foundation which focused on financial area, education and healthcare respectively.

3.2.2 Decision-making techniques

A variety of decision-making methods have applied into many research areas recently. The popular selected decision-making methods are presented as below:

Analytic hierarchy process (AHP): is a well-known theory of measurement through pairwise comparisons and relies on the judgements of experts to derive priority scales (Saaty 2008). Maˇcek and Alagi´c (Maček & Alagić 2017) described a AHP model to evaluate Bitcoin cryptocurrency in the context of information security risks related to the existing most common online payment systems like

Table 3.1: Key requirements of blockchain-based internet mapped to existing blockchain elements

Decision-making methods	**Domain specific**	**References**
benchmarking	Not Defined	(Keary 2012, Wurster et al. 2018, Bach et al. 2018)
AHP	financial area, education, and healthcare; e-banking, m-banking, e-commerce	(Viriyasitavat & Hoonsopon 2018, Dorri, Kanhere & Jurdak 2017)
Fuzzy based	Supply chain	(Hileman & Rauchs 2017, Çolak et al. 2020)
TOPSIS	Healthcare, Internet of Vehicle	(Liu et al. 2020, Rathee et al. 2020)

e-banking, m-banking, and e-commerce. Fuzzy-based methods: uses linguistic variables to express the comparative judgments given by decision makers, and linguistic variables are expressed qualitatively by linguistic terms and quantitatively by a fuzzy set in the universe of discourse and respective membership function (Kahraman et al. 2003). Ferhat et al. (Karayazi & Bereketli 2020) proposed a multi-criteria decision model to assist a global logistics company on the blockchain software selection problem using Buckley's Fuzzy Analytical Hierarchy Process (Fuzzy AHP). Technique for Order Preference by Similarity to Ideal Solution (TOPSIS): defined the positive ideal matrix and negative ideal matrix and calculated the distance between expert and the decision matrix to determine the weights of decision makers (Shih et al. 2007). Mohammad et al. presented an assessment method to evaluate the impact of different blockchain models using TPOSIS for healthcare management.

Table 3.1 summarized the selected studies that discuss the blockchain platform selection problem. The studies using benchmarking are based on documentations and reports, which are often out of date soon and should keep updating continuously. And the majority of decision model using AHP, Fuzzy based and TOPSIS are appropriate for specific case studies. However, it may not be suitable for knowledge-based conversation applications. In addition, the current decision model always focused on the selection of various decision-making methods and choose the most efficient one applied into domain area, but no matter which decision-making method selected, it still has some limitations and choosing different methods may have inconsistent results. It is better to use multiple measurements to solve conflict and get consistent evaluation result. In our proposal, we propose a standardized framework to guide decision-making process for blockchain-based conversation, which can be used for keeping updating, and a decision model with multiple measurements will be used to evaluate different blockchain platforms to select the best fitting blockchain platform.

3.3 Modeling Decision-making steps for blockchain platforms

3.3.1 Requirement analysis

The conversation system is composed of multiple experts who can share and construct a knowledge database collaboratively to address many complex tasks efficiently. The user's request may be related to different scenarios and these questions should be solved by multiple experts based on a predefined cooperation strategy. Thus, the knowledge-based conversation system is an integrated system for implementing conversation scenario management, decision support, and intelligent search, which has characteristics

shown in Table 3.2 .

Table 3.2: The requirement analysis of the knowledge-based conversation system

Key requirements	Detailed description	Design Aim
Content reliability and confidentiality	Content may be related to the user, e.g., personal information, so that conversational content maintained by each participating expert should be kept secure and be identified clearly. This enables participants to make their contributions in a trustworthy and secure manner (Zheng, Ma & Wang 2017, Qin et al. 2016). Expert knowledge should also be protected during sharing.	1) Identifier content should have secure storage. 2) Shared knowledge should be transferred and stored safely. 3) Contributions should be assigned without central control that can reduce the possibility of insecurity caused by human factors.
Immediacy and accurate response	Responses should be immediate and users' requests should be replied to accurately(Calvaresi et al. 2019).	1) Agreement between multiple experts should be reached to get a consensus. 2) Support fast searching and matching in the knowledge database to get an accurate response.
Open-ended and extensible	The more open the system is, the more efficient the knowledge base, which should also expand in the future (Shibata et al. 2009, Tang et al. 2019).	1) Design a fair incentive scheme to encourage communities of participants to cooperate and create value. 2) The handling capacity of the system should have a certain redundancy to expand the functional module.

Based on the analysis shown in Table 3.2, we can see that the main requirements of the conversation system include content reliability and confidentiality, immediacy response and open-ended and extensible.

Content reliability and confidentiality: Conversation content is very important for users to protect their privacy and for each expert to evaluate their contributions during sharing and maintenance (Henderson et al. 2018, CHEN et al. 2017). Thus, this content should be clearly identified and ensure data integrity. The design aims including security storage, security knowledge sharing and contributions assignment without central control are identified to meet this requirement.

Immediacy and accurate response: to solve tasks, multiple experts should find the best solutions based on an efficient consensus mechanism. This assurance can allow the system to reach an agreement over the whole network and to have global collaboration between multiple experts. Current knowledge-based conversation systems lack consensus methods among multiple experts to support immediate and accurate responses (Zheng, Ma & Wang 2017, Qin et al. 2016). Therefore, reaching agreement between multiple experts as well as fast searching and matching knowledge base are design aims to satisfy immediate and accurate response.

Open-ended and extensible: to support multiple experts sharing and maintaining knowledge base together, an incentive scheme must be built to encourage multiple experts to promote good content and restrict bad content. And the knowledge base should keep expanding in the future. Based on these requirements, a fair incentive scheme and the capacity of expansion are necessary design aims.

3.3.2 Modeling Concepts

Based on the requirement analysis of a knowledge-based conversation system, we modeled the selection of blockchain platform as a decision-making problem. The modeling of decision-making steps for blockchain platform selection are shown as Figure 3.1.

As shown in Figure 3.1, based on the identified design aims, we can formulate the modelling concepts into three steps. First, decision items are selected according to the design aims. Then, the multiple evaluating criteria are identified for decision items. Finally, applicability levels of blockchain applied in conversation systems are assigned based the opinions from experts or related documentations.

3.3.2.1 Decision items for evaluating blockchain platforms

Based on the requirement analysis of knowledge-based conversation system, four group decision items, including decentralized architecture, storage and sharing, computing performance, and scalability are selected to match our design aim, then the detailed items are identified as well. Then for each group of decision item, corresponding blockchain configurations are chosen as evaluation criteria for evaluating blockchain platforms.

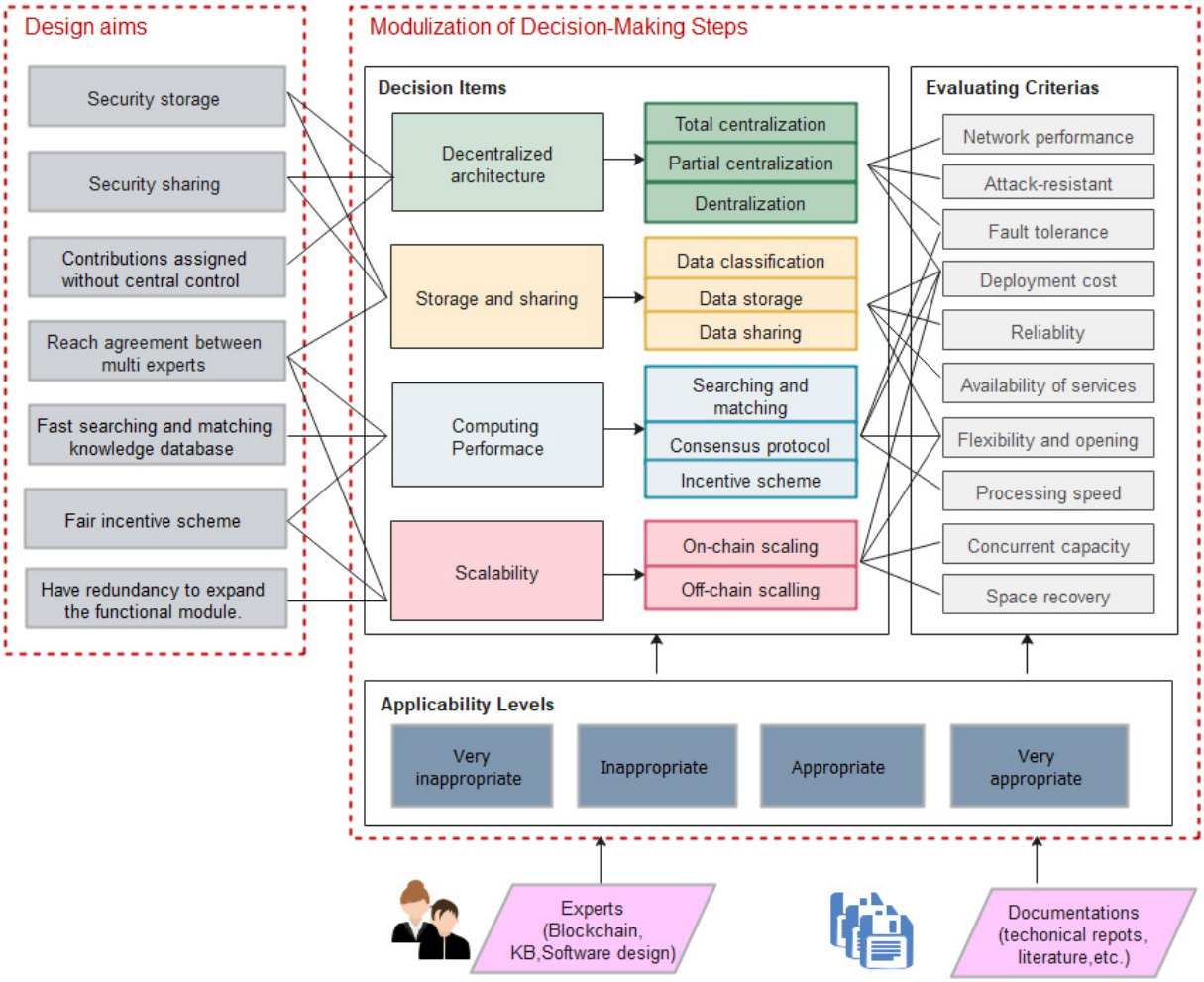


Figure 3.1: The overall modeling concepts.

Decentralized Architecture: there are three architectures regarding trust mechanism: centralized, partial centralized and total decentralized (Rahmadika et al. 2018). The centralized architecture is relied on a single or a few entities that control the entire network, and the partial centralized architecture is used to manage authority allocation of nodes, such as read-write access, transaction authority, etc. The decentralized architecture is based on decentralized transaction and data management which can provide anonymity, safety, and data integrity. There is no need of a third-party organization to control the transactions, making this field a vast research area to deal with limitations and enhancements in the centralized and partial architectures. Different architectures may cause different blockchain types and chain structures. Public chain is fully decentralized where any node can join and leave the system with better transaction transparency and security but affect the network performance. The consortium chain provides the data sharing between organizations, while the private chain is used to manage access permission within single organization.

Storage and Sharing: The content in the knowledge-based conversation system includes different types of scenarios and permissions so that how to design the storage and sharing mode is also an important point. Blockchain has two storage modes: the on-chain and off-chain. On-chain is used to storage metadata, key data and hash values, while off-chain can be considered as private cloud or third-party storage (Eberhardt & Tai 2017, Hepp et al. 2018). In addition, the contents may be related to identical information and unimportant information, we should decide whether it should be stored permanently or not. Thus, it not only considers the reliability and availability of service, but also considers flexibility and how to reduce the deployment cost.

Computing Performance: Regarding the computing and processing in the knowledge-based conversation system, it needs to consider the blockchain configuration such as consensus protocol, incentive scheme, and to decide if we need to design the new method or not (Bach et al. 2018). Consensus algorithms are the core technology for blockchain, since it determines who keeps records, which would influence the security and reliability for the whole system. So far, there have been many consensus algorithms, with Proof of Work, Proof of Stake, and Delegated Proof of Stake, the most popular ones. However, we should consider the processing speed and fault tolerance of protocols, and if there needs to be some improvements based on the existing methods, the deployment cost and flexibility should be also considered.

Scalability: To support variety conversation scenario and multi-expert decision, the blockchain configurations should consider on-chain scaling and off-chain scaling, many on-chain scaling such as Segregated Witness and off-chain scaling such as Lightning Network are used to increase blockchain size and storage efficiency (Chauhan et al. 2018, Kim et al. 2018). Different blockchain platforms use different scaling methods including space recovery, parallel verification, deployment cost, etc.

3.3.2.2 Levels of applicability

According to the satisfaction degree matched with design aims, applicability levels are defined in Table 3.3 as very inappropriate, inappropriate, appropriate, and very appropriate.

3.3.2.3 Formulation the blockchain platform selection with multiple criteria

For each group of decision item, the main flow is presented to formulate the selection of blockchain platform with multiple criteria based on expert opinions and literature reviews. The arrows are used to

Table 3.3: Key requirements of blockchain-based internet mapped to existing blockchain elements

Applicability levels	Applicability statement
Very inappropriate	Does not satisfy the design aim using blockchain-based mechanisms.
Inappropriate	Satisfies the design aim using blockchain-based mechanisms but only fewer blockchain configurations are acceptable.
Appropriate	Satisfies the design aim using blockchain-based mechanisms and most blockchain configurations are acceptable.
Very appropriate	Satisfies the design aim using blockchain-based mechanisms and almost all the blockchain configurations have a high performance.

represent the possible order for decision making.

The design flow starts from deciding whether it is needed to use decentralization architecture or not. Total centralization architecture has a higher network performance and is easy to deploy but has lower attack-resistance that provides attackers with a single target to hack. Partial centralization also has appropriate network performance, attack-resistance, and fault tolerance because of limited node access and transactional authority to perform refinement operations. Decentralized architecture has the highest attack-resistance but lowest network performance, and it is also hard to deploy compared to centralization because it is open to the public (Rahmadika et al. 2018).

Table 3.4: Design decisions regarding decentralization architecture

Decision item	Description	Network performance	Deployment cost benefits	Attack-resistant	Fault tolerance
Total centralization	Single/multipoint service	++++	++++	+	+++
Partial centralization	Permission chain	+++	+++	+++	+++
Decentralization	Public chain	++	++	++++	++

+: very inappropriate, ++: inappropriate, +++: appropriate, ++++: very appropriate.

The second design decision is the on-chain and off-chain division regarding the data classification, storage and sharing. The on-chain mechanism can use transaction constraints and smart contracts to provide storage and sharing, which is more reliable. Transaction constraints have some limitations on transaction type and size, while a smart contract uses variables and log events to support more flexible storage and sharing. At present, Bitcoin only provides simple on-chain storage, while the functions in Ethereum are more powerful with smart contracts. The off-chain mechanism can use the local private cloud or a third-party platform, which is easy to deploy and supports the flexible availability of the service but has low reliability because it is easily erased or alterable without a trace (Hepp et al. 2018).

Then, the design decisions regarding computing performance include searching and matching,

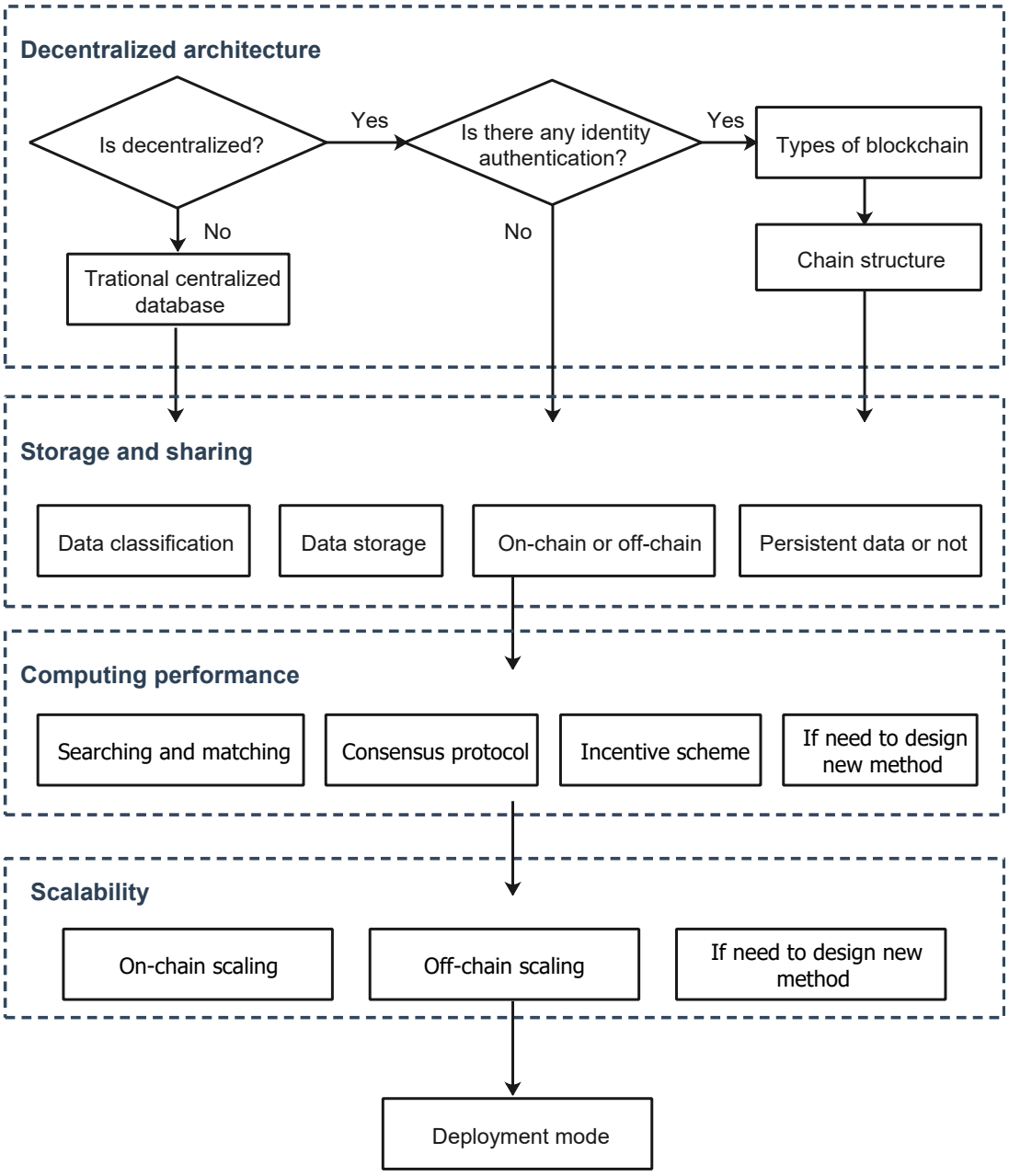


Figure 3.2: The main workflow to guide formulation models.

consensus protocol, incentive scheme will be processed. Searching and matching are related to the chain structure. Single chain has lower processing speeds and flexibility but is easy to deploy, while side chain and multi-chain have high processing speeds to support more conversation scenarios. The consensus protocols can be divided into proof-based and voting-based. The agreement base of proof-based consensus algorithms, such as PoW and PoS, is following nodes to perform enough proof, and most nodes can join freely. It always has high decentralized and low processing speeds, while voting-based verifies the network from a majority of node decisions under limited executing nodes, such as Paxos and practical Byzantine fault tolerance (PBFT), which have higher processing speeds and fault tolerance.

Table 3.5: Design decisions regarding storage and sharing

Decision item	Description		Reliability	Availability of service	Flexibility and opening	Deployment cost benefits
Data classification	On-chain	Transaction constraints	++++	++	++	++
		Smart contract		+++		
	Off-chain	Localization/ third-party platform	+++	+++	+++	++++
Data storage	On-chain	Embedded transaction (Bitcoin)	++++	+	+	++
		Embedded transaction (Ethereum)		++	+	++
		Smart contract		+++	+++	+++
	Off-chain	Localization	++	+++	+++	++++
		Third-party platform		+++	+++	+++
Data sharing	On-chain	Transaction constraints	++++	++	+++	++
		Smart contract		+++		
	Off-chain	Localization, third-party platform	+	+++	+++	++++

+: very inappropriate, ++: inappropriate, +++: appropriate, ++++: very appropriate.

Additionally, the incentive system includes an economic-based model and a non-economic model that compensates individuals with money or other rewards. The token economic model is easy to compute with high speed but has less flexibility and stability. The non-economic model is not related to financial factors and for the conversation system, the design of the non-economic model should focus on how to provide participants with more opportunities to receive good content, which may need more flexible computing processes(Bach et al. 2018).

With more experts and uses being involved, the popularity has brought the network scalability problem to light. There are two ways to scale a system to handle millions of transactions: on-chain scaling and off-chain scaling. With on-chain scaling, all transactions are made in the blockchain. At present, one way is to increase the blocksize directly to process more transactions in a short time. Another way is to remove the overhead from the block to increase data storage. These methods can improve the little space available; however, it is still an utterly inadequate method of dealing with the scalable problem.

Table 3.6: Design decisions regarding computing performance

Decision item	Description	Processing speed	Deployment cost benefits	Flexibility and opening	Fault tolerance
Searching and matching	Single chain	++	++++	++	++
	Side chain	+++	+++	+++	+++
	Multi-chain	++++	+++	++++	+++
Consensus protocol	Proof-based	++	++	+++	++
	Voting-based	+++	+++	+++	+++
Incentive scheme	Economic model	+++	++	++	+++
	Non-economic model	++	+++	++++	+++

+: very inappropriate, ++: inappropriate, +++: appropriate, ++++: very appropriate.

Off-chain scaling uses extra layers on top of the blockchain to deal with most of the transactions, which will be bundled as one and saved on the blockchain, such as the Lightning Network. This can create channels across peers using a two-layer network without any limitations in blocksize. Compared to on-chain scaling, off-chain scaling is more flexible and has better concurrency (Kim et al. 2018).

Table 3.7: Design decisions regarding scalability

Decision item	Description	Deployment cost benefits	Flexibility and opening	Concurrent capacity	Space recovery
On-chain scaling	Increasing the blocksize	++	++	++	++
	Removing the overhead	+++	+++	++	++
	Shading	+++	+++	++	++
Off-chain scaling	Lightning Network	+++	+++	+++	+++
	Raiden Network	+++	+++	+++	+++
	Plasma	+++	+++	+++	+++

+: very inappropriate, ++: inappropriate, +++: appropriate, ++++: very appropriate.

3.4 A Decision model for blockchain platform selection

In general, a decision model contains three basic layers: target, criteria, and alternatives as shown in Figure 3.3. For our research, the target is to analyze and select the most suitable blockchain platform for knowledge-based conversation system. Criteria are used to make decisions based on the identified decision items. The alternative platform options to be evaluated are Ethereum, Fabric, Corda, Multichain

and even more.

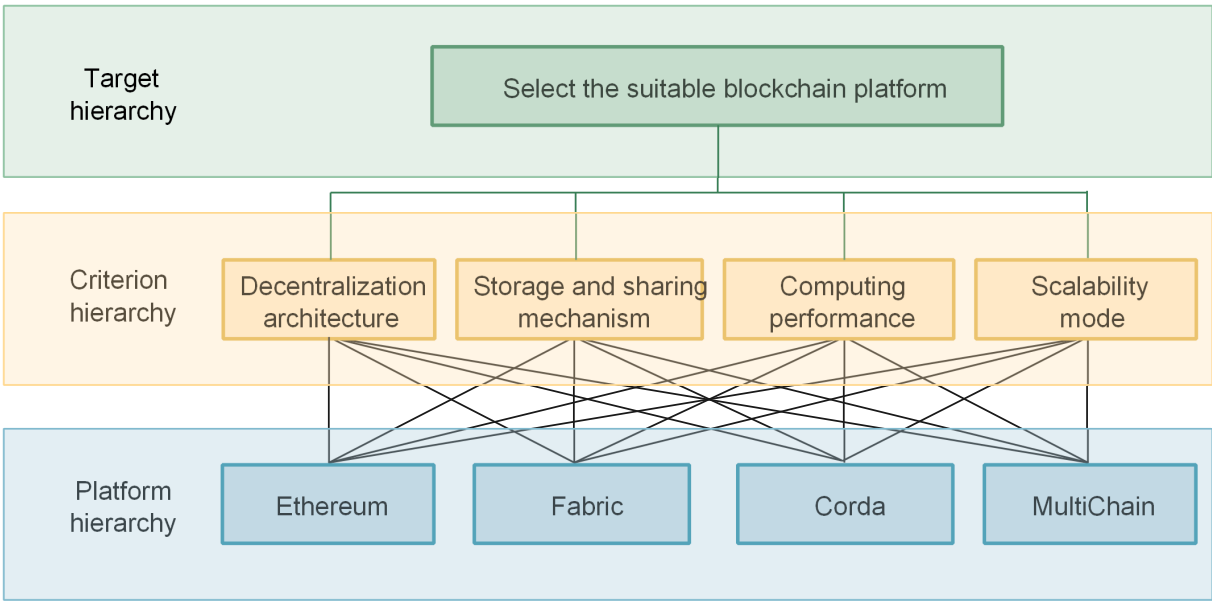


Figure 3.3: Basic Hierarchy model of process design.

In our proposed decision model, first, weighted membership matrixes of each blockchain platform are built by using multiple criteria. Then the evaluation results are composed and synthesized using multiple measurements. Finally, the final decision result will be judged by consistent checking and get the best fitting blockchain platform.

3.4.1 Weighted Membership Matrix

Based on the established model, the multiple evaluation criteria for a blockchain-based conversation system is presented in Table 3.8.

Based on the evaluation criteria selected in Table 3.8, we set the evaluation criteria U= {U1,U2,U3,U4,U5,U6} as below to build the membership matrix for each alternative blockchain platform.

- Access authority: private chain, consortium chain, public chain.
- Chain structure: single chain, side chain, multi-chain.
- Storage and sharing mechanism: on-chain and off-chain.
- Consensus protocol: PoW, PoS, Byzantine fault tolerance (BFT), PBFT.
- Incentive scheme: token, no financial factors, both.
- Scale mode: on-chain, off-chain, others.

Table 3.8: Evaluation criteria and sub-criteria for a blockchain-based conversation system

Criterion	Sub-criterion	Detailed items
Decentralization architecture	Access authority	Private chain, Consortium chain, Public chain
	Chain structure	Single chain, Side chain, Multi-chain
Storage and sharing	Data classification	On-chain, Off-chain
	Data storage	On-chain, Off-chain
	Data sharing	On-chain, Off-chain
Computing performance	Consensus protocol	Proof-based: PoW, PoS, Voting-based: PBFT, Raft, Others: Notary
	Incentive scheme	Token economic model, Non-economic model with no financial factors, Both
Scalability	Scale mode	On-chain: increasing blocksize, removing overhead, shading. Off-chain: Lightning Network, Raiden Network, Plasma.

Then experts' opinions are collected to identify the criterion and sub-criterion using DELPHI method with a 0.1–0.9 rating scale (Lin & Liao 2017)

3.4.2 A weighting method using multiple measurements for decision making

In this section, to overcome the limitations and inconsistent decisions making by single measurement, we aim to identify the consistent criteria weighting for selecting blockchain platforms and rank such platforms using multiple measurement methods. We will compare the results from these three methods and provide the most appropriate options for blockchain platform selection. The detailed steps are introduced how to use these three methods, respectively.

3.4.2.1 Multiple measurement methods

In our paper, the three most popular weighting methods including AHP, Fuzzy based AHP and TOPSIS are utilized to combine and evaluate the criteria weighting.

1) Analytical hierarchy process (AHP)

The analytical hierarchy process is a multiple criteria decision-making method that was presented by Prof. Thomas L. Saaty (Saaty 2008). AHP simplifies preference ratings in decision criteria using pairwise comparisons. By checking the consistency of attribute values during measurements, AHP can eliminate bias and conflicts in decision making. Using AHP normally involves four main steps:

Step 1: Decomposing the decision-making problem into a hierarchy structure with general levels.

Step 2: Developing a pairwise comparison matrix, establishing priorities between criteria in the hierarchy using the nine-point scale presented by Saaty and Vargas (Goepel 2019), and normalizing the resulting matrix.

Step 3: Synthesizing judgments to calculate percentages for weight attributes, which includes normalizing the comparison matrix and computing the weights.

Step 4: Calculating the consistency ratio to measure the above judgments, which are consistent, and obtaining the set of final weights. The consistency criteria (CI) is calculated by CI = (λmax-n)/n-1, where λmax is the maximum eigenvalue of the judgment's matrix, and the consistency ratio is CR=CI/RI.

2) Fuzzy analytical hierarchy process (FAHP)

The FAHP method is an updated analytical method developed from the classic AHP. It was difficult to set uncertain attributes from the crisp values using AHP; therefore, FAHP was proposed to resolve the uncertainty of the AHP approach by performing fuzzy comparisons, which makes decisions for multiple criteria using weight derivations from a fuzzy pairwise comparison. Chang (1992) proposed a creative algorithm for dealing with pairwise comparison scales by using triangular fuzzy numbers. In 1996, he introduced a new analysis algorithm for simulated values of pairwise comparisons. So far, FAHP has been used to make decisions, such as the selection of cryptographic algorithms for blockchain, evaluating the risks of blockchain, and other issues. The process of FAHP for blockchain platform selection has the following four steps (Aydin & Kahraman 2013):

Step 1: Building the evaluation hierarchy structure for selecting total n blockchain platforms.

Step 2: Determining the evaluation dimension weights using a 0.1–0.9 scale to build the judgment matrix $A = (a_{ij})_{n\times n}$.

Step 3: Establishing the fuzzy consistent matrix $R = (r_{ij})_{n\times n}$. whose elements have degrees of membership.

$$r_{ij} = \frac{r_i - r_j}{2n} + 0.5, \text{ where } r_i = \sum_{i=1}^{n} a_{ij}, i = 1, 2, \ldots n \tag{3.1}$$

Step 4: Calculating the weight vector of the elements in all dimensions of the hierarchy system by using the root-squaring method.

$$W^{(0)} = \left[\frac{\sqrt[n]{\prod_{j=1}^{n} r_{1,j}}}{\sum_{j=1}^{n} \sqrt[n]{\prod_{j=1}^{n} r_{i,j}}}, \frac{\sqrt[n]{\prod_{j=}^{n} r_{2,j}}}{\sum_{j=1}^{n} \sqrt[n]{\prod_{j=1}^{n} r_{i,j}}}, \ldots \frac{\sqrt[n]{\prod_{j=1}^{n} r_{n,j}}}{\sum_{j=1}^{n} \sqrt[n]{\prod_{j=1}^{n} r_{i,j}}} \right] \tag{3.2}$$

3) Technique for Order Preference by Similarity to Ideal Solution (TOPSIS)

TOPSIS is a popular multiple attribute decision-making (MADM) method first proposed by Hwang and Yoon in 1981, further refined by Yoon in 1987, and then updated again by Hwang et al. (1993). TOPSIS is a type of compensatory aggregation method that compares all substitutes by determining weights for every attribute, normalizing the scores for every attribute, and calculating the geometric distance between each alternative and the ideal alternative. The ideal alternative is the one with the best score in every attribute. The attributes of TOPSIS alternatives are assumed to be monotonically increasing or decreasing. Trade-offs could exist between TOPSIS attributes, which means that a wrong result in any one attribute could be denied by a correct result in any other attribute [38].

The short TOPSIS process for our BaaS selection is based on the following steps:

Step 1: Take m alternatives and n criteria to create an evaluation matrix, using $A = (a_{ij})_{n\times n}$ to get the intersection of each criterion and alternative.

Step 2: The matrix, A, is then normalized to form a matrix.

$$B = (b_{ij})_{n\times n} = a_{ij}/\sqrt{\sum_{i=1}^{n} a_{ij}^2}, i, j= 1, 2, \ldots n \tag{3.3}$$

Step 3: Define the weight of each criterion in the evaluation matrix: $w= [w_1, w_2, \ldots w_n]$, then get the weighting normalized matrix $C = (c_{ij})_{n*n}$.

Step 4: Determine the worst alternative $C^-$and the best alternative.

$$C_j^+ = \begin{cases} \max c_{ij}, \text{where criteria } j \text{ is positive criteria} \\ \min c_{ij}, \text{where criteria } j \text{ is negative criteria} \end{cases}$$

$$C_j^- = \begin{cases} \min c_{ij}, \text{where criteria } j \text{ is positive criteria} \\ \max c_{ij}, \text{where criteria } j \text{ is negative criteria} \end{cases} \tag{3.4}$$

Step 5: Calculate the L2-norm distances D^- and D^+between the target alternative, i and the worst condition, C^-, and the distance between the alternative, i, and the best condition, C^+, respectively.

$$D_i^+ = \sqrt{\sum_{j=1}^{n} (c_{ij} - C_j^+)^2}, D_i^- = \sqrt{\sum_{j=1}^{n} (c_{ij} - C_j^-)^2} \tag{3.5}$$

Step 6: Calculate the similarity to the worst condition then rank the alternatives according to the results.

3.4.2.2 Composing and synthesizing

Based on the membership matrix R=$\{R_e, R_f, R_c, R_m\}$ of blockchain platforms as well as calculated weighting W=$\{W_{AHP}, W_{FAHP}, W_{TOPSIS}\}$ from three measurements, the weighted average method by $\mathrm{B} = \mathrm{W}^{\circ}\mathrm{R}$ is used to get the three groups of evaluation results for each alternative blockchain platforms:

$\{B_{e_{\mathrm{AHP}}}, B_{f_{\mathrm{AHP}}}, B_{c_{\mathrm{AHP}}}, B_{m_{\mathrm{AHP}}}\}$,

$\{B_{e_{\mathrm{FAHP}}}, B_{f_{\mathrm{FAHP}}}, B_{c_{\mathrm{FAHP}}}, B_{m_{\mathrm{FAHP}}}\}$

$\{B_{e_{\mathrm{TOPSIS}}}, B_{f_{\mathrm{TOPSIS}}}, B_{c_{\mathrm{TOPSIS}}}, B_{m_{\mathrm{TOPSIS}}}\}$.

3.4.2.3 Consistent Checking

Based on the above results, Eq (3.6) was used to judge and determine the selection of blockchain platforms.

$$\mathrm{W} = \sum_{i=1}^{4} (i^{\circ} B_i) \tag{3.6}$$

When $|\mathrm{W} - \mathrm{n}| \leq 0.5, (\mathrm{n} = 1, 2, 3, 4)$, the final evaluation level will be defined in the *n* level. The results with same level using all three methods will be finalized as the consistent result, which can be candidates to select the most suitable blockchain platforms.

3.5 Evaluation results and analysis

3.5.1 Implementation stages

From the decision model described in section 4, the following stages were conducted in the research.

Stage 1. Determine the weighted membership matrix of each alternative blockchain platform.

- Collect the experts' opinions to identify the criterion and sub-criterion using expert DELPHI method with 0.1-0.9 rating scale.
- Build the membership matrix of blockchain platforms based on their technical features.

Stage 2 Implementation of AHP for weighting decisions and ranking blockchain platforms.

- Build the comparative matrix of attributes based on hierarchical levels of the sub-criterion.
- Calculate the consistency ratio and verify AHP consistency.
- Get the relative attribute weights of each sub-criterion.

Stage 3 Implementation of FAHP for weighting decisions and ranking blockchain platforms.

- Build the comparative matrix of attributes based on hierarchical levels of the sub-criterion.
- Build the fuzzy consistent matrix.
- Get the fuzzy synthesis values.
- Calculate the attribute weights of each sub-criterion.

Stage 4 Implementation of TOPSIS for weighting decisions and ranking blockchain platforms.

- Build the comparative matrix of attributes based on hierarchical levels of the sub-criterion.
- Normalize the membership matrix.
- Establish the worst alternative and best alternative.
- Calculate the similarity value and rank the alternatives' scores.

Stage 5 Compose and synthesize to evaluate the results of each blockchain platform.

- Calculate the averaged weighting of each blockchain platform by using AHP, FAFP, TOPSIS respectively.

Stage 6 Consistent checking to determine the evaluation results.

- Get the final level of each blockchain platform by using AHP, FAFP, TOPSIS respectively.
- Choose candidates with same levels.
- Determine the best fitting decision with the highest level in the candidates.

3.5.2 Research limitations

The main limitations of this research can be regarded as the obtaining opinion of insufficient experts, and they may lack knowledge to determine the applicability of all the criterions. Furthermore, the selection of designing factors in this study is based on the requirement analysis of multi-expert conversation system, and it is possible to consider and include more criterions as well as sub-criterions.

3.5.3 Evaluation Results

We presented the experimental results regarding weighted membership matrix, AHP, FAHP, TOPSIS, composing and synthesizing as well as consistent checking as below:

Results regarding the weighted membership matrix

For building the AHP, FAHP, and TOPSIS, the opinions of ten artificial intelligence experts were obtained and used for applicability in evaluating each criterion and sub-criterion. We used a 0.1–0.9 rating scale based on the expert DELPHI method using four levels: very inappropriate, inappropriate, appropriate, and very appropriate. The results of all criteria regarding the membership matrix are shown in Table 7.

Based on the results of applicability for all criteria, we can build the membership matrix of Ethereum, Fabric, Corda, and MultiChain according to the technical features of these blockchain platforms.

Ethereum is a decentralized platform using public chain. Its storage and sharing mechanism are both on-chain. It is used to build a single-chain structure. The consensus protocol of Ethereum is PoW and the incentive scheme is a token-based model. It can use the Raiden Network or Plasma to reduce network congestion and facilitate the speed of processing.

$$R_e = \begin{bmatrix} 0.0\ 0.2\ 0.2\ 0.6 \\ 0.6\ 0.2\ 0.1\ 0.1 \\ 0.3\ 0.2\ 0.2\ 0.3 \\ 0.9\ 0.1\ 0.0\ 0.0 \\ 0.3\ 0.2\ 0.2\ 0.3 \\ 0.0\ 0.1\ 0.2\ 0.7 \end{bmatrix}$$

Table 3.9: The results of applicability levels for all criteria

Criterion	Sub-criterion	Very inappropriate	Inappropriate	Appropriate	Very appropriate
U1	Private chain	0.5	0.3	0.1	0.1
	Consortium chain	0.1	0.2	0.2	0.5
	Public chain	0.0	0.2	0.2	0.6
U2	Single chain	0.6	0.2	0.1	0.1
	Side chain	0.4	0.4	0.2	0.2
	Multi-chain	0.1	0.1	0.2	0.6
U3	On-chain	0.3	0.2	0.2	0.3
	Off-chain	0.0	0.0	0.8	0.2
U4	PoW	0.9	0.1	0.0	0.0
	PoS	0.6	0.2	0.1	0.0
	PBFT	0.0	0.2	0.3	0.5
	Raft	0.0	0.2	0.2	0.6
	Notary	0.1	0.2	0.2	0.5
U5	Token	0.3	0.2	0.2	0.3
	Non-financial	0.0	0.3	0.3	0.4
	Both	0.0	0.0	0.1	0.9
U6	On-chain	0.1	0.2	0.2	0.5
	Off-chain	0.0	0.1	0.2	0.7

Fabric is partial centralization architecture with consortium chains and its original transaction and computing are used off-chain (InterPlanetary file system) for storage. The consensus protocol also supports multi-chain. Fabric can provide a second layer solution to allow for off-chain scalability using both Fabric token and user-defined non-economic incentive schemes.

$$R_f = \begin{bmatrix} 0.1\ 0.2\ 0.2\ 0.5 \\ 0.1\ 0.1\ 0.2\ 0.6 \\ 0.0\ 0.0\ 0.8\ 0.2 \\ 0.0\ 0.2\ 0.2\ 0.6 \\ 0.0\ 0.0\ 0.1\ 0.9 \\ 0.0\ 0.1\ 0.2\ 0.7 \end{bmatrix}$$

Corda is also a consortium blockchain platform based on partial centralization architecture and it supports off-chain computing. The multi-chain structure can be also designed in Corda. It has several notary clusters and each cluster can be deployed to a different consensus algorithm. We do not suggest building a digital currency or token when using Corda as it supports layer two to scale blockchain.

$$R_c = \begin{bmatrix} 0.1\ 0.2\ 0.2\ 0.5 \\ 0.1\ 0.1\ 0.2\ 0.6 \\ 0.0\ 0.0\ 0.8\ 0.2 \\ 0.1\ 0.2\ 0.2\ 0.5 \\ 0.0\ 0.3\ 0.3\ 0.4 \\ 0.0\ 0.1\ 0.2\ 0.7 \end{bmatrix}$$

MultiChain is a platform that helps users build certain private chains used by organizations. The consensus protocol is PoW. Multichain is compatible with Bitcoin, which also uses on-chain storage and sharing, and supports many different tokens, including Lightning Network.

$$R_m = \begin{bmatrix} 0.5\ 0.3\ 0.1\ 0.1 \\ 0.1\ 0.1\ 0.2\ 0.6 \\ 0.3\ 0.2\ 0.2\ 0.3 \\ 0.9\ 0.1\ 0.0\ 0.0 \\ 0.3\ 0.2\ 0.2\ 0.3 \\ 0.1\ 0.2\ 0.2\ 0.5 \end{bmatrix}$$

Results regarding the implementation of AHP methodology

Step 1: Building the comparative matrix of attributes

Based on the proposed AHP process, we constructed our selection hierarchy architecture of blockchain platforms as shown in Table 3.9. According to the requirement analysis of the knowledge-based conversation system collected from experts, the comparative matrix of attributes was established, as shown in Table 3.10, according to the nine-point scale.

Table 3.10: Comparative judgment matrix for criterions

Blockchain platforms	Access authority	Chain structure	Storage and sharing	Consensus	Incentive	Scale mode
Access authority	1	1/4	1/3	1/5	3	2
Chain structure	4	1	1/2	1/3	4	3
Storage and sharing	3	2	1	1/4	4	3
Consensus	5	3	4	1	7	6
Incentive	1/3	1/4	1/4	1/7	1	1/2
Scale mode	1/2	1/3	1/3	1/6	2	1

Step 2: Calculating the consistency ratio and verify AHP consistency

To check the consistency of the matrix, first, we calculated the largest eigenvalue $\boldsymbol{\lambda}_{max}$ of the comparative matrix. Then, Eq (3.7) was used to calculate the $\mathrm{CI} = 0.0621$ and $\mathrm{CR} = 0.0501$, with the random-generated consistency index, as shown in Table 3.11.

$$\mathrm{CI} = \frac{(\boldsymbol{\lambda}_{max} - n)}{(n-1)} = \frac{(6.3106 - 6)}{(6-1)} = 0.0621, \mathrm{CR} = \frac{CI}{RI} = \ 0.0501 < 0.1 \tag{3.7}$$

Table 3.11: Random-generated consistency index

n	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49

If CR$<$0.1, then the comparative matrix processes a better consistency. Otherwise, we need to adjust the comparative matrix processes a satisfied consistency.

Step 3: Get the relative attribute weights

The relative contribution of each attribute to the target is determined by calculations made using the Eigenvector V= (0.1565, 0.3276,0.3732, 0.8427,0.0762,0.1139). Then each attribute weight is $w_i = v_k / \sum_{i=1}^{k} v_k,\ where\ k = 6$ and the final attribute weight is $W_{AHP} = (w_1, w_2, \ldots w_k)$. The result

is W_{AHP}=$(0.0828, 0.1733, 0.1975, 0.4458, 0.0403, 0.0602)^T$.

Results regarding the implementation of FAHP methodology

Step 1: Building the comparative matrix of attributes

This was the same as the AHP method and we established the comparative matrix of attributes as shown in Table 3.10.

Step 2: Building the fuzzy consistent matrix

The original comparative matrix of experts' opinions was converted to a fuzzy consistent matrix using triangular fuzzy number. Table 3.12 shows the results of the fuzzy consistent matrix for the criteria.

Step 3: Find the sum of every lowest value (L), middle value (M) and Upper value U values to be fuzzy synthesis values for triangular fuzzy number.

Step 4: Get the normalized weight calculation after the comparison of fuzzy synthesis values (see Table 3.13) using the following degree of possibility calculation.

Then we normalize of vector weight to get the final attribute weights.

$$V\,(M2{\geq}M1) = \begin{cases} 1 & ,if\ m_2{\geq}m_1 \\ 0, & ,if\quad l_1{\geq}u_2 \\ \frac{l_1-u_2}{(m_2-u_2)-(m_1-l_1)^r}, & etc \end{cases} \tag{3.8}$$

W_{FAHP}=$(0.0885, 0.2128, 0.1452, 0.3311, 0.0981\ 0.1243)^T$.

Table 3.12: Fuzzy consistent matrix for criteria

Blockchain platforms	Access authority	Chain structure	Storage and sharing	Consensus	Incentive
Access authority	[1, 1, 1]	[0.5000, 0.6667, 1]	[0.6667, 1, 1.5000]	[0.4000, 0.5000, 0.6667]	[0.6667, 1, 1.5000]
Chain structure	[1, 1.5000, 2]	[1, 1, 1]	[1, 1.3333, 2]	[0.6667, 1, 1.5000]	[1, 1.5000, 2]
Storage and sharing	[0.6667, 1, 1.5000]	[0.5000, 0.7500, 1]	[1, 1, 1]	[0.5000, 0.6667, 1]	[1, 1.5000, 2]
Consensus	[1.5000, 2, 2.5000]	[0.6667, 1, 1.5000]	[1, 1.5000, 2]	[1, 1, 1]	[2.5000, 3, 3.5000]
Incentive	[0.6667, 1, 1.5000]	[0.5000, 0.6667, 1]	[0.5000, 0.6667, 1]	[0.2857, 0.3333, 0.4000]	[1, 1, 1]
Scale mode	[1, 1.3333, 2]	[0.6667, 1, 1.5000]	[0.6667, 1, 1.5000]	[0.3333, 0.4000, 0.5000]	[0.5000, 0.7500, 1]

Table 3.13: The L, M and U values for triangular fuzzy number

Blockchain platforms	L	M	U
Access authority	0.0710	0.1240	0.2209
Chain structure	0.1015	0.1850	0.3313
Storage and sharing	0.0824	0.1492	0.2650
Consensus	0.1649	0.2774	0.4472
Incentive	0.0752	0.1261	0.2286
Scale mode	0.0793	0.1383	0.2485

Results regarding the implementation of TOPSIS methodology

Step 1: Building the comparative matrix of attributes

This was the same as the AHP and FAHP methods and we established the comparative matrix of attributes as shown in Table 8.

Step 2: Normalizing the membership matrix

We established the normalized evaluation membership matrix for all criteria, as shown in Table 3.14.

Table 3.14: The normalized matrix for criteria

Blockchain platforms	Access authority	Chain structure	Storage and sharing	Consensus	Incentive	Scale mode
Access authority	0.1395	0.0663	0.0796	0.1780	0.3078	0.2598
Chain structure	0.5581	0.2650	0.1194	0.2967	0.4104	0.3897
Storage and sharing	0.4186	0.5301	0.2388	0.2226	0.4104	0.3897
Consensus	0.6977	0.7951	0.9552	0.8902	0.7182	0.7795
Incentive	0.0465	0.0663	0.0597	0.1272	0.1026	0.0650
Scale mode	0.0698	0.0883	0.0796	0.1484	0.2052	0.1299

Step 3: Establishing the worst alternative and best alternative

Based on the above weighted normalized value, the highest and lowest values were considered as the best and worst solutions for each criterion.

Best solution $C^{+}= (0.6977, 0.7951,\ 0.9552,\ 0.8902,\ 0.7182,\ 0.7795)$

Worst solution $C^{-}= (0.1395, 0.0663,\ 0.0597,\ 0.1272,\ 0.1026,\ 0.0650)$

Step 4: Calculating the similarity value and ranking the alternatives' scores

The distance of each normalized weighted value from the best and worst solutions was calculated according to Eq (3.5). Then rank the alternatives' scores were determined by the sum of the distance values from the best and worst solutions. Finally, we normalized the alternatives' scores and obtained the

final result:

$$W_{FTOPSIS}=(0.0799, 0.1836,\ 0.1999, 0.5013,\ 0,\ 0.0353)^T.$$

Results regarding the implementation of Composing and synthesizing

By using AHP, FAHP and TOPSIS, the evaluation results of Ethereum, Fabric, Corda and Multichain are calculated by weighted average method respectively as below:

Table 3.15: The evaluation results by using three measurements

Blockchain Platforms	AHP	FAHP	TOPSIS
B_e	$[0.58,\ 0.15,\ 0.09, 0.18]$	[0.50,0.15,0.11,0.23]	[0.62,0.15,0.08,0.15]
B_f	$[0.03,\ 0.13,\ 0.31, 0.53]$	[0.03,0.12,0.28,0.58]	[0.03,0.14,0.32,0.52]
B_c	$[0.07,\ 0.14,\ 0.32, 0.47]$	[0.06,0.15,0.30,0.49]	[0.08,0.14,0.32,0.47]
B_m	$[0.54,\ 0.15,\ 0.10, 0.21]$	[0.45,0.15,0.12,0.27]	[0.57,0.14,0.09,0.20]

Results regarding the implementation of Consistent Checking

Based on the above calculated results, the Eq (3.7) is used to judge and determine the selection of blockchain platforms, and the final evaluation levels will be show in Table 3.16.

Table 3.16: The candidates with same evaluated levels

Blockchain Platforms	AHP	FAHP	TOPSIS
B_e	1.87	2.05	1.76
B_f	3.34	3.39	3.35
B_c	3.19	3.22	3.2
B_m	1.98	2.19	1.92

Thus, based on the above results, Ethereum and Multi-chain gets all same level 2 while Fabric and Corda gets all same level 3. Both can be candidates to be selected.

3.6 Discussion and Summary

Based on the results of AHP, FAHP, and FTPOSIS, the comparison of the final weight for each criterion is shown in Figure 3.4. In short, the results summarized in Table 16 are consistent with the rankings of AHP and FTOPSIS; however, there are some differences between the obtained results by AHP and FAHP, and these differences can be explained by the following points:

- The calculation mechanism was different between AHP and FAHP and FTOPSIS. In classical AHP and FTOPSIS, the numerical values of variables are used for evaluating criteria; however, in the FAHP method, the decision-making of criteria was determined by fuzzy numbers.

- In classical AHP, the consistency process is used to measure the judgments, while fuzzy AHP does not require any consistency mechanism because of fuzziness.

- The characteristic of evaluations is another factor. Since probable deviation is used to integrate the decision-making process in FAHP, the evaluation results are a more natural process considering the uncertain characteristics of information, compared to the AHP method.

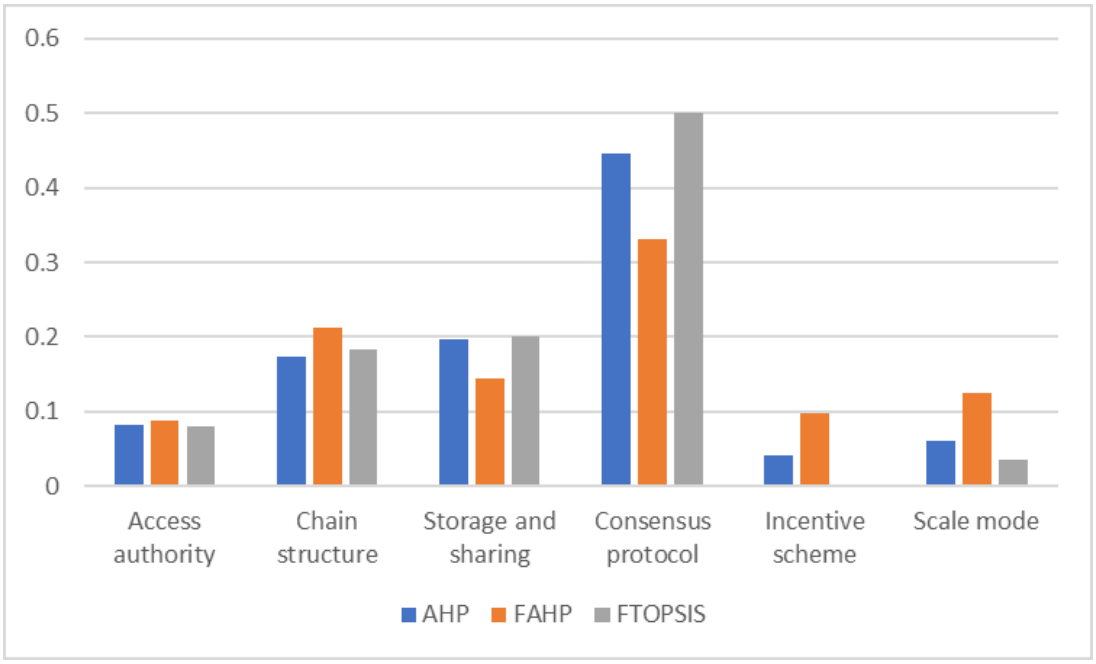


Figure 3.4: Comparison of the final weight for each criterion using AHP, FAHP, and FTOPSIS.

In summary, from comparing the final weights for each criterion using AHP, FAHP, and FTOPSIS, it seems that the top three criteria are consistent with these three methods. Hence, the consensus protocol, chain structure, and storage and sharing are the most important considerations when selecting blockchain platforms for a conversation system.

Furthermore, based on the evaluation results show in Table 3.17, there are no differences between the rankings of alternatives using these three methods. The results show that Hyperledger Fabric is the first choice for use in a conversation system compared to Ethereum, Corda, and Multichain. However, the blockchain platform can also be changed according to future requirements.

For the conversation system, the domain knowledge maintenance is a collaborative process by multiple experts, which will be built through selected blockchain platform to make our conversation system more efficiently. Based on the above discussion and analysis, the Hyperledger Fabric blockchain ledger is used to establish as a knowledge base for our conversation system. To implement this, the following components can be explored in the future works. First, the Fabric multi-chain structure can

Table 3.17: The results of the difference between rankings of criteria

Criterions	AHP	FAHP	FTOPSIS	$d_{AHP\text{-}FAHP}$	$d_{AHP\text{-}FTOPSIS}$
Access authority	4	6	4	-2	0
Chain structure	3	2	3	$+1$	0
Storage and sharing	2	3	2	-1	0
Consensus protocol	1	1	1	0	0
Incentive scheme	6	5	6	$+1$	0
Scale mode	5	4	5	$+1$	0

be used to store different domain knowledge data with various membership, which provides flexible functions to identify authorities. Second, knowledge rules and afterwards can be generated and validated through blockchain smart contracts to guarantee transparency and non-tampering. Last but not the least, the reward scheme based on expert reputation can be utilized to motivate experts for knowledge base maintenance. In this way, the knowledge base can be implemented as a Fabric wrapper and can support more reliable conversation service.

Chapter 4. Master–slave blockchain in conversation system

Conversation systems always involves multi-domain conversation interactions that increases rapidly because more domains are considered. Therefore, multiple experts must maintain these conversation interactions, which cause experts from different domains to be unable to interact properly. Consequently, ensuring secure and efficient cooperation between different experts has become a crucial problem that needs to be solved. The experimental results show that the proposed blockchain structure is feasible and effective for handling different domains in a conversation system.

4.1 Introduction

Blockchains are based on distributed ledger technology (DLT), which is spread across several nodes or computing devices (Nakamoto n.d.), aiming to provide a trustworthy service without a central authority. Thus far, a majority of blockchain applications, such as financial (Cosares et al. 2021) and supply-chain (Dujak & Sajter 2019) blockchains, still support simple transactions on a single blockchain, which lowers the consensus performance of the blockchain with increasing workloads or scaling. Further, many application areas require diversified digital assets, which can handle more complex transactions; however, a single blockchain can only support transaction verification and storage by traversing all of the data, which greatly reduces the performance.

Considering the above mentioned drawbacks of using a single blockchain, some researchers have proposed blockchain interoperability to handle diversified digital assets. One such technique has been used for communicating between the master chain and slave chain (Johnson et al. 2019) . Further, Shaoyong et al. proposed a master–slave blockchain in IoT to achieve cross-domain authentication (Guo et al. 2020*a*), and Zhaofeng et al. presented a master–slave blockchain for digital-rights management DRM-related applications (Ma et al. 2018). A grid terminal data security management model based on

the master–slave blockchain was proposed by Zhengwen et al. (Zhang et al. n.d.). Another technique involves using a multichain for inter-blockchain communication (Kan et al. 2018); Ashar et al. proposed a scalable and efficient multichain solution for auditing applications based on the practical Byzantine fault tolerance (PBFT) protocol (Kong et al. 2019). Further, Aida et al. described the implementation of a private blockchain using the multichain open-source platform, with potential applications in food tracking and tracing, product lifecycle management, and counterfeit prevention (Ismailisufi et al. 2020). However, interoperable blockchains vary depending on the requirements. The above-mentioned chain model cannot be used directly in a conversation domain.

4.2 Related work

In this section, we provide an overview of state-of-art research in two primary areas regarding blockchain interoperability as well as threat model.

4.2.1 Blockchain interoperability

With exponential increases in the use of blockchain technology, blockchain interoperability is becoming a crucial subject for enhancing communication between blockchains. For our conversation system, since conversation interactions could pass through multiple domains, connecting and sharing the content of these conversations in different domains is important. Interoperability between blockchains can smoothly connect and share information in a trustworthy manner. Thus far, several solutions have been proposed to enable the exchange of various digital assets across different blockchains. Here, we will describe the main chain model with the six currently used for blockchain interoperability, and compare their performances, as shown in Table 1.

Sidechain model: These models are master-slave blockchains that allow the bi-directional transfer of assets between the master chain and the slave chain at a fixed or pre-determined exchange rate. Slave chains can have their own protocols and implementation methods, differing completely from those of the master blockchain. The master-slave structure is built with an overlay scheme (Singh et al. 2020)(Guo et al. 2020*b*), acting as the master overlay to handle the intercommunication for multiple slave overlays.

- **RSK**: this was created to be compatible with Ethereum's applications, but uses Bitcoin as the underlying cryptocurrency (Lerner 2015). It has a two-way relationship with the Bitcoin blockchain and rewards Bitcoin miners via merged mining with Bitcoin or any other blockchain, sharing the

Bitcoin block format and proof-of-work.

- **Ardor**: this is a blockchain platform that uses a unique parent–child chain architecture (*Ardor White-paper* 2017). Child chains are separate blockchains with their own native tokens; for different use cases in different domain, they are integrated into the parent chain. Ardor uses a 100% proof of stake (PoS) consensus algorithm.

- **Liquid**: this is proposed as a sidechain-based settlement network used in cryptocurrency exchanges. It is built on the Bitcoin codebase that allows users of the Liquid Network to move Bitcoin between the two distributed networks using the Liquid Proof of Stake (LPoS) consensus protocol (Nick et al. 2020).

Multichain model: these models run using multiple parallel blockchains. Chains are completely independent of each other, supporting diversified digital assets. Different chains can communicate with each other, but they do not ensure that global data is consistent.

- **Cosmos**: This allows multiple parallel blockchains to run, which can communicate with each other via protocols like virtual UDP or TCP, and each independent blockchain is powered by the BFT consensus algorithms (Abraham et al. 2017).

- **Polkadot**: This is a scalable multi-chain framework that enables cross-blockchain transfers of any type of data or asset. Each separated blockchain is implemented using a different segment of the Polkadot network, and is based on the nominated proof-of-stake to validate proofs (Wood 2016).

- **Multichain**: This is designed for interoperability among private blockchains (Greenspan 2015). MultiChain can work with different blockchains simultaneously and provide customized privacy and control within the same network. Multichain applies handshaking to connect nodes between different chains, along with proof of authority (PoA) consensus for permissioned blockchains.

Based on the discussion above and the requirements of multi-domain conversation systems, it is clear that sidechain solutions such as RSK, Ardor and Liquid, can satisfy the requirements of security and data consistency. However, they limit the number of stakes and ignore many important aspects in conversation systems, such as reputation and contribution. While multichain model solutions like Cosmos, Polkadot and Multichain can support different blockchains running in parallel and improve the transaction throughput, the chains are independent of each other. However, this affects the consistency of data. Therefore, a suitable sidechain model needs to be designed for multi-domain conversation systems.

Table 4.1: Comparison of the existing block interoperability solutions

Blockchain	Consensus	Advantages	Drawbacks	References
RSK and Ardor	PoS	Fully trusted. Can ensure consistent data.	Flow energy efficiency. Favours rich nodes, as the selection limits the number of tokens owned by the node.	(Lerner 2015) (*Ardor Whitepaper* 2017)
Liquid	LPoS	Highly energy efficient in comparison to pure PoS, since it adds optional delegation with voting rights. Fully trusted Can ensure consistent data.	Favours rich nodes, as the selection limits the number of tokens that the node owns. The number of delegators is technically limited by the minimum requirements of bond size.	(Nick et al. 2020) (Ahmad et al. 2019)
Cosmos	BFT	Highly energy efficient, as a limited number of validators are required through the voting process.	Semi-trusted, because nodes are not subject to loss of stake even they are voted as a bad validator. Cannot ensure the consistency of global data.	(Abraham et al. 2017) (Kan et al. 2018) (Kwon & Buchman 2018)
Polkadot	NPoS	Highly energy efficient, as they nominate a set of validators on the network with their DOT. Does not require special hardware.	Semi-trusted, as all DOT nodes are controlled by a proof of stake consensus network in a centralized manner. Cannot ensure the consistency of global data.	(Wood 2016) (Cevallos & Stewart 2020)
Multichain	PoA	Highly energy efficient, as it has a limited number of validators owing to selection rules. No strict need for a native token on the blockchain to reward validators.	Semi-trusted, as the notion of data immutability is lost based on blacklists and censorship measures. Cannot ensure the consistency of global data.	(Greenspan 2015) (De Angelis et al. 2018)

4.2.2 Threat model

Similar to single blockchain communication, some common attacks that affect cross-chain communication include denial-of-service (DoS) attacks and transaction-based injection attacks (Bissias et al. 2016) (Vokerla et al. 2019). Here, we list three attack strategies that our system may encounter:

- A malicious node transmits incorrect and malicious messages that can damage the system.
- A malicious device could request multiple identities in our system, thereby crashing the network and inducing network redundancy.
- A malicious node can pretend to be an honest node to gain a higher reputation, and then attack the system.
- The motivation of attackers is to obtain more rewards. In the consensus mechanism, robustness against such attacks is important.

4.3 Master–slave blockchain for multi-domain conversation systems

4.3.1 Overall architecture

According to the background elucidated in section 4.2, the overall architecture of the multi-domain conversation system is based on a master-slave blockchain based on Hyperledger Fabric 2.0 platform, as shown in Fig. 1. To support transaction concurrency among different domains in a trusted manner, the architecture has two layers. One layer comprises slave chains with different conversation domains. Each conversation domain contains multiple domain experts to maintain the conversation content. The other layer is a master chain, which is used to realize inter-chain authentication to access conversation content in multiple domains and store the summary of intra-chain certification to maintain the global consistency of transactions.

Based on the architecture shown in Figure 4.1, five types of chain nodes can be seen: the slave node, endorsement node, leader node, anchor node and master node. The function of each type of node is introduced as follows:

- Master node: is node in the master chain, which will be leader node, endorsement node, anchor node

as well as committer node at the same time. The master node verifies the inter-chain authentication and stores the summary of intra-chain certification in the master chain. Master nodes have digital signatures signed with their private keys.

- Slave node: is used to reach a consensus and maintain the content of different domain chains. Slave nodes have digital signatures signed with their public keys.

- Leader node: is used to communicate with the master chain, which is responsible for submitting inter-chain authentication to request verification, as well as sending the hash of intra-chain certification to be stored in the master chain. The domain slave node with the highest number of votes will be selected as the leader node.

- Endorsement node: is used to verify the transactions before performing the consensus algorithm, and also simulates the execution of the smart contract according to the endorsement policy and return the result with respective certificate signature to the application client. Each slave chain can have more than one endorsement nodes, which normally the nodes with higher reputations.

- Anchor node: is responsible for inter-chain communication. Inter-chain contents need to make agreement with cross-domain slave nodes and achor nodes will transfer the contents through different slave chains when some nodes can access verified by master nodes. Then they can work together to make cross-chain consensus. The anchor nodes need to setup considering stability, because anchor nodes have any problems, then the cross-domain communication will be break off.

4.3.2 Data structure based on hash anchoring

The master-slave chain has one master chain and multiple slave chains. The master chain is constructed by verification blocks, while the slave chain comprises of domain blocks. The data structure of the master–slave chain is based on hash anchoring shown in Figure 4.2.

The verification block is maintained in the entire network by master nodes, while the domain block contains details of intra-chain transactions. Different slave chains can utilize different data formats according to the requirement from different domains. Specifically, the verification blocks in the master chain will store a summary of domain blocks within a time frame, as well as inter-chain certifications during this time frame. Domain blocks in the slave chain have intra-chain certifications, including conversation contents as well as contribution assignments.

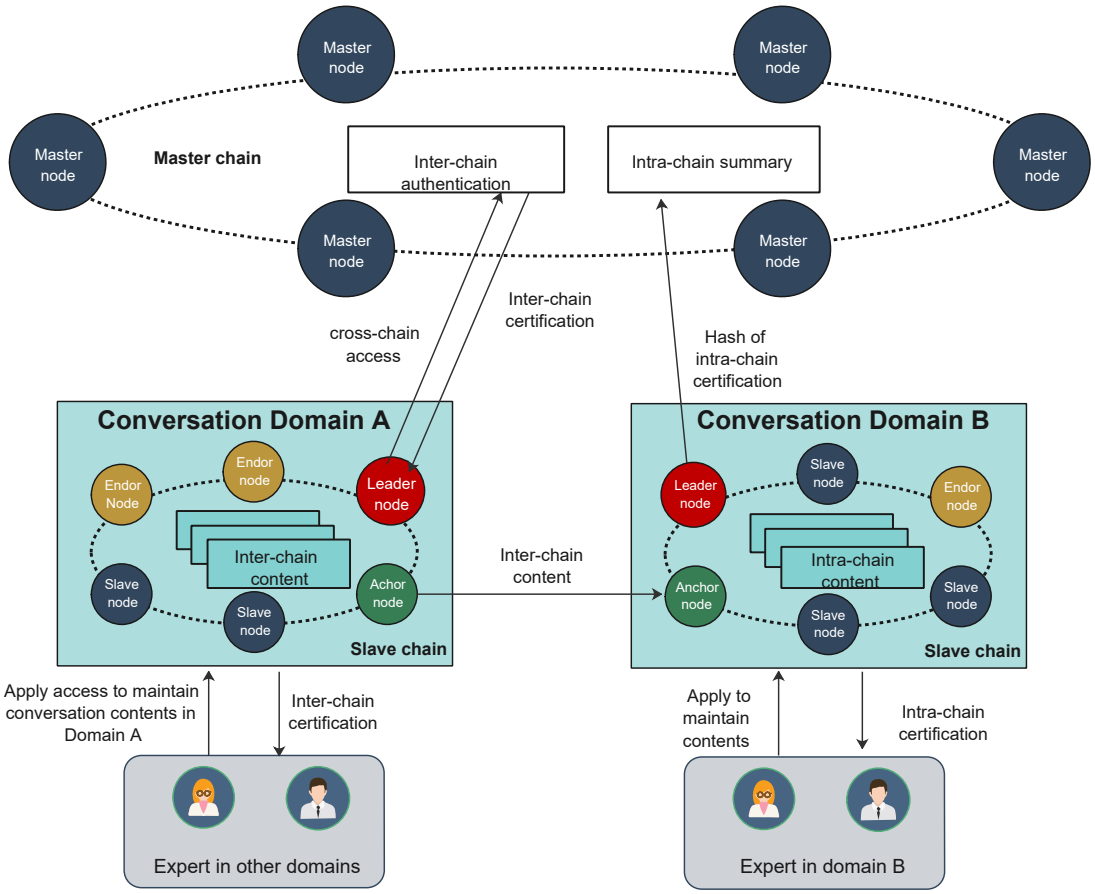


Figure 4.1: Overall architecture based on master–slave chain model.

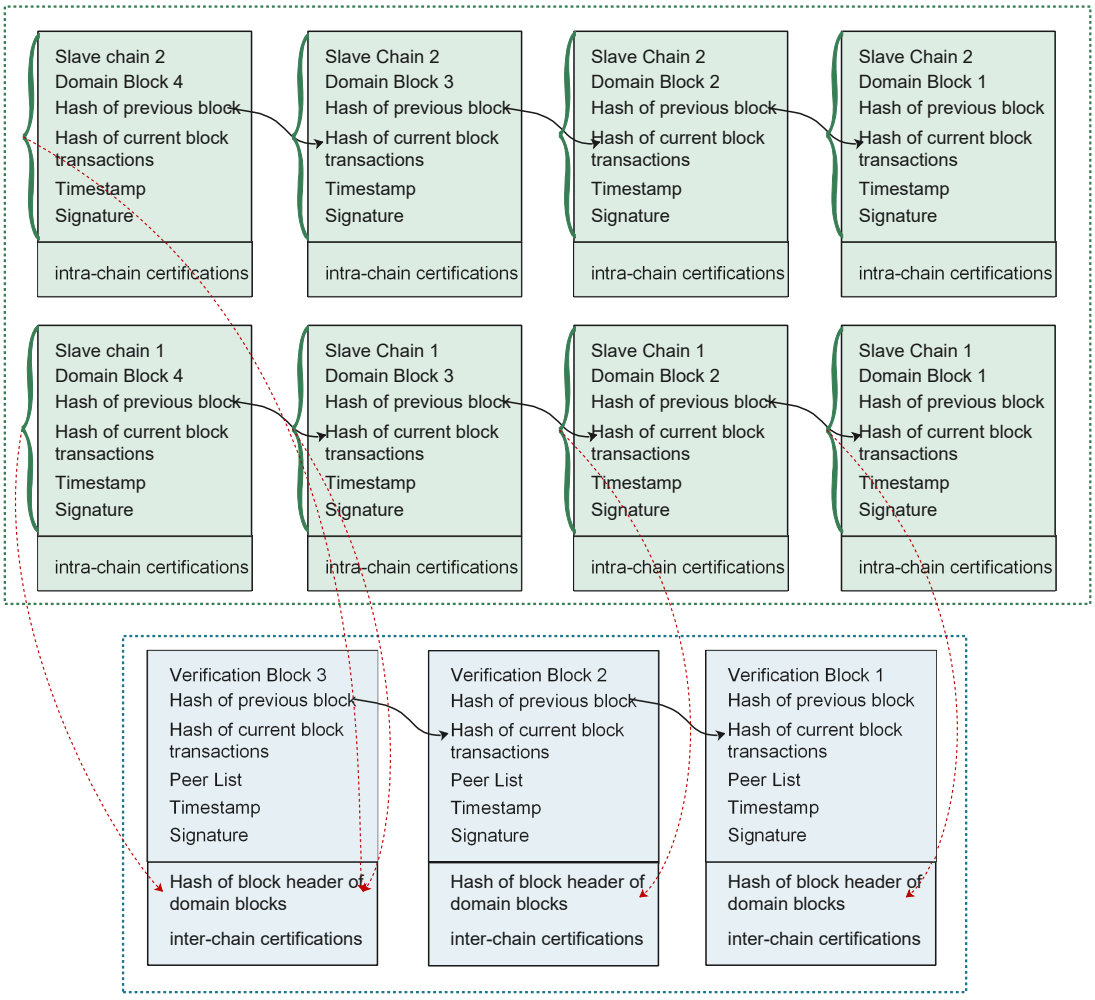


Figure 4.2: Data structure of master–slave chain.

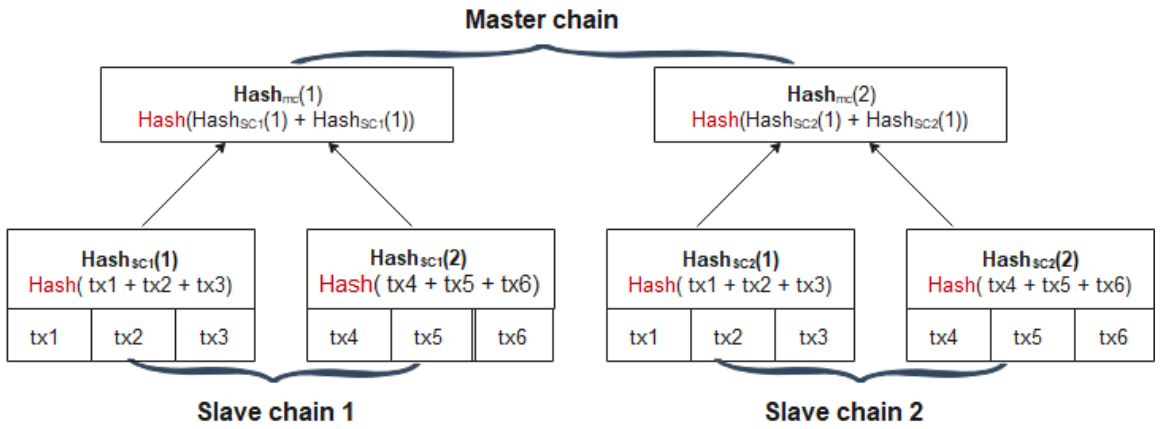


Figure 4.3: Calculation example for summarizing domain blocks.

To guarantee the master–slave chain structure is not tampered, hash anchoring is used to link the master chain and slave chains. In the body of the verification block, the summaries of domain blocks are stored. The sample summary of domain-block headers is shown in Figure 4.3.

Based on the presented calculation example, the hashes of the domain-block's transactions in different slave chains are represented as $Hash_{sc1}(i)$ and $Hash_{sc2}(i)$. Then, the summary of domain blocks in the master chain is calculated as $Hash_{mc}\,(i)$. It can be seen that any changes of transactions in the domain blocks will also need to alter the master chain, which means if attacks change any contents in slave chains, master chain can find this change and present tampering. In this way, it can guarantee the tamper-proof nature of the blockchain technology.

4.4 Evaluation and Discussion

In this section, we will evaluate the performance of the proposed master-slave chain, including proof of Feasibility and transaction throughout in order to verify the applicability of the proposed master–slave chain structure.

4.4.1 Proof of Feasibility

Based on the proposed master–slave chain, we now provide proof of the feasibility of how the proposed blockchain structure keeps the data consistent and prevents threats. To achieve this, here we assume the consensus algorithm is using default Raft consensus protocol and we set the owner of block *B* as O_B, and φ_B is the confirmation time to add or discard a block.

Theorem 1: This algorithm can prevent 50% of the attacks on the master chain. Assume that M_B is a master node, and create a block *B* with illegal data. Our consensus algorithm should end up with all nodes that discard block *B*, thus 50% of the attacks will fail.

Proof 1: We assume the number of fault-tolerance nodes is M and the total number of nodes is N. As long as $M < (N-1)/3$, the probability of a malicious block becoming a normal block is always less than 50%. Moreover, this happens when all fault-tolerance nodes become the master nodes evaluated using Eq.(3.1), and the probability *P* of the malicious nodes all becoming master nodes is:

$$P = \lim_{M \to \infty} \left(\frac{M}{N} \right)^M \tag{4.1}$$

It can be seen from Eq.(4.1) that P will approach 0, and 50% of the attacks will fail.

This proves that our proposed master-slave chain can prevent 50% of the attack on the master chain.

Theorem 2: This algorithm prevents malicious voting in the slave chain. Assume that S_B is a slave node and send a malicious transaction that can reduce the reputation value of other nodes or increase its reputation value. The proposed master-slave chain should prevent malicious activities.

Proof 2: In our proposed blockchain structure, all the domain blocks in the slave chain will have related block summaries stored in the master chain. Here, we use cost calculation to guarantee the prevention of malicious; one portion of the cost is related to tampering with the domain block content in the slave chain, and the other portion is related to the cost when tampering with the master block and its related block summary in the master chain. Assume that the height of malicious block *B* in the slave chain is $H_{sc}(B)$, the current block height in the slave chain is H_{sc}, and N_{sc} is the number of domain nodes in the slave chain network. The current block height in the master chain is H_{mc} and N_{mc} is the number of master nodes in the master chain network. Then, the minimum number of blocks required to tamper with each node is:

$$B_{min} = (H_{sc} - H_{sc}(B)) * N_{sc} + (H_{mc} - H_{mc}(B)) * N_{mc} \tag{4.2}$$

As shown in Eq.(4.2), if S_B sends a malicious transaction to change the reputation value in the slave chain, they need to tamper with B_{min} blocks at the minimum. The cost for this is large, and this cost will only increase with an increasing number of domain nodes and master nodes.

From a probabilistic perspective, as shown in Eq.(4.1) in Proof 1, the probability of obtaining

creation rights for a malicious domain node is:

$$P_B = (\frac{1}{2})^M \lim_{B_{min}\to\infty} \left(\frac{M}{N}\right)^{B_{min}} \tag{4.3}$$

From Eq.(4.2) and Eq.(4.3), along with the increasing number of domain nodes *M and* master nodes *N,* the number of blocks B_{min} that must be tampered with also increases, and P_B approaches zero.

Based on the above mentioned theorems and proofs, it is clear that the proposed master-slave chain is feasible to keep data consistent and prevent possible attacks.

4.4.2 Blockchain Deployment

As we analyzed in Chapter 3, the knowledge base can be implemented as a Fabric wrapper and can support more reliable conversation service. Here we deploy Hyperledger Blockchain 2.0 into Ali Cloud, which has one master chain and three slave chains with one orderer service, three organizations and total 100 nodes. We deployed in standalone server to simulate the multiple nodes using different port numbers. The specifications of the standalone server are shown in Table 4.2.

The deployment network is shown in Table 4.3 and the sample structure of Hyperledger Fabric network is shown as Figure 4.4. Appendices C shows the sample network configuration of master chain with three organizations and total 9 peers, while three slave chains are deployed using the similar structure that each slave chain has two organizations and 30 peers. Here, we generate and deploy one chaincode with single smart contract to update reputation of each peer.

The experimental transactions are simulated by created peers,and the constructed transactions include both inter-chain and intra-chain transactions. The inter-chain transaction are used for updating reputation of peers that belong to the same slave chain, while intra-chain transactions are used for backup the summary of inter-chain transactions. The total number of experimental transactions are 5000 (4000 inter-chain transactions and 1000 intra-chain transactions). We set the maximum transaction number of each block is 100, the block size is 512KB and the consensus time threshold is 10 seconds.

Table 4.2: Specifications of the stand-alone server

Item	Specification
Operation system	Ubuntu20.04
Processor	8CPU 16GHz
Hard Drive Space	600 GB for 64-bit CentOS
JDK	8
Internet link speed	144 (Mbps)

Table 4.3: The three group tests

Type		**Node**
Test 1:Single domain		
Single chain		40
Master-slave chain (one slave chain)	Master chain	10
	Slave chain 1	30
Test 2:Two domains		
Single chain		50
Master-slave chain (two slave chains)	Master chain	10
	Slave chain 1	30
	Slave chain 2	30
Test 3:Three domains		
Single chain		100
Master-slave chain (three slave chains)	Master chain	10
	Slave chain 1	30
	Slave chain 2	30
	Slave chain 3	30

4.4.3 Experimental results and discussion

To verify the efficiency of the proposed master–slave chains, in our experiments, we compared the experimental results using raft in both single chain and the proposed master-slave with same number of nodes. The test scenarios are defined as below:

- stand-alone server, single domain dataset.
- stand-alone server, two domain dataset.
- stand-alone server, three domain dataset.

We analyzed the three group tests shown in Table 4.3 to discuss the transaction throughout.

The transaction throughput results are tested based on the above scheduling and shown in Figure 4.5 and 4.6, respectively.

Based on the experimental results, there are no obvious difference between single domain and multiple domains of single blockchain model, while for the master-slave chain model, it can be concluded

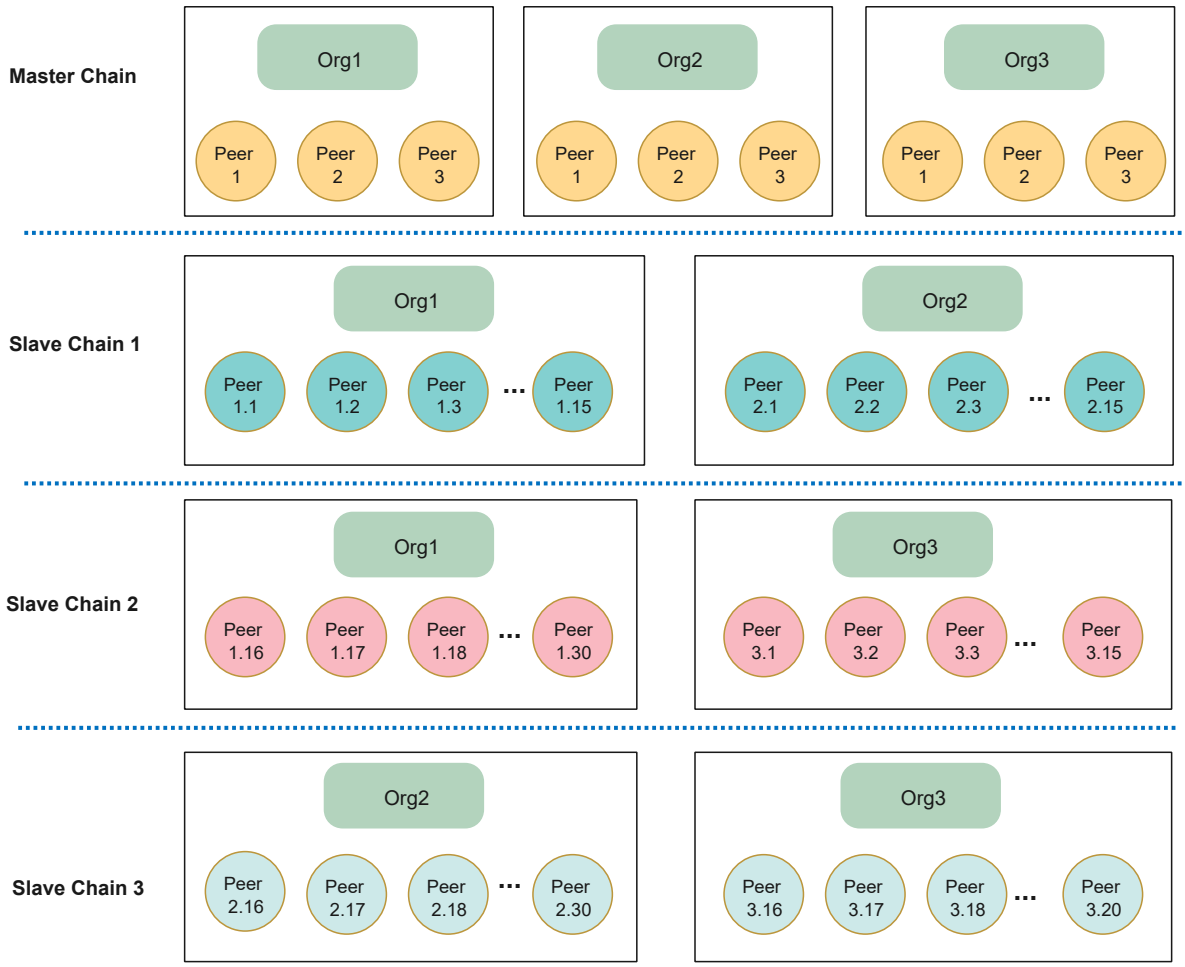


Figure 4.4: The sample deployment structure of Hyperledger Fabric.

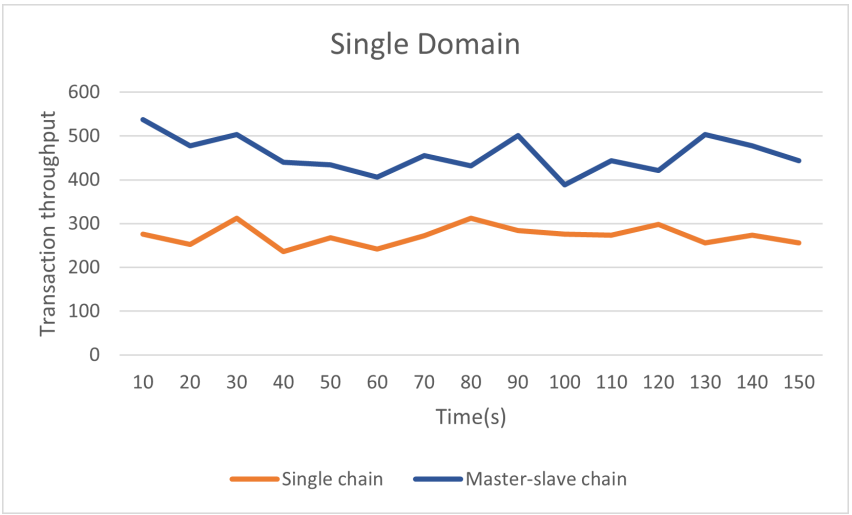


Figure 4.5: The comparison results of transaction throughput between single chain and master-slave chain in single domain

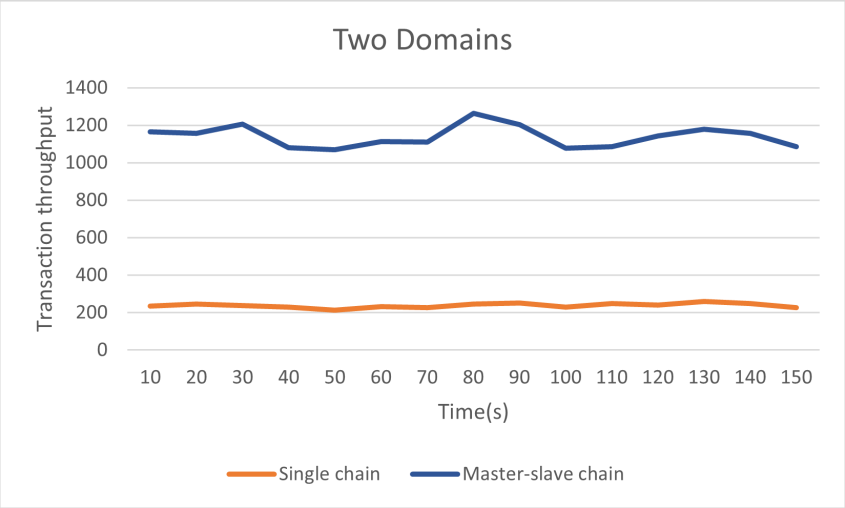


Figure 4.6: The comparison results of transaction throughput between single chain and master-slave chain in two domains.

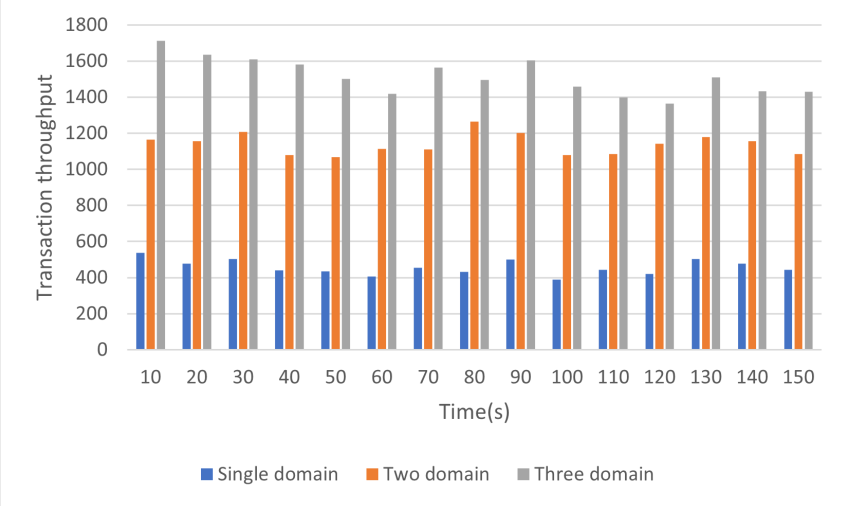


Figure 4.7: The comparison results of transaction throughput between single domain, two domains and three domain for master-slave chain

that the transaction throughput is related to the number of slave chains with different domains, that is: the more slave chains, the greater the transaction throughput performance. Thus, the proposed master-slave chain model has been confirmed that concurrently generating blocks and processing different types of domain transactions is an effective way to improve the throughput of blockchain transactions.

We can see the Raft in master-slave chain has the advantages of high throughput and fast confirmation. However, in the default Raft consensus algorithm, the leader selection and voting strategy is based on random election timeout, which may not be secure and efficient way to make sure quality of conversation contents. For example, the node with minimum election timeout will request vote and may grant as leader node, but if this leader node doesn't have enough reputation and experience, it may reduce the

satisfaction degree of our conversation system. In addition, the endorsement and anchor nodes in original Raft algorithm will be setup when deploy Fabric network,which can not be dynamic changed. This may cause secure and effective issues as well, because the reputation of these nodes may decreased along with the development of conversational services. Thus, the fairness and dynamics of consensus algorithm for conversation system still need to improve.

4.5 Summary

In this chapter, we proposed a master–slave chain model to process multiple conversation interactions concurrently from different domains, and we evaluated the proposed mechanism from both theoretical proofs and experiments to verify its feasibility and efficiency. Furthermore, the consensus algorithm in Hyperledger Fabric used for transaction verification still has some security and efficiency issues that need to be improved, which we will discuss in the chapter 5.

Chapter 5. Hybrid consensus algorithm for master–slave blockchain

Based on the proposed master-slave chain, considering the efficiency and security of the conversation system, a hybrid consensus mechanism is proposed in this chapter. This mechanism is suitable for our designed master-slave chain for a conversation system. First, a consensus based on reputation-driven voting is utilized for intra-chain verification. Second, a dynamic construction strategy is used to select the master consensus nodes for inter-chain authentication. Furthermore, an incentive scheme is designed to generate both economic and non-economic rewards for all the nodes participating in the consensus process. The evaluation results show that the proposed consensus mechanism is effective for all experimental scenarios.

5.1 Introduction

As a fundamental and key component of blockchain technology, consensus algorithms comprise a set of rules and procedures to ensure equality and fairness, and to maintain and validate a set of data among the participating nodes (Nguyen & Kim 2018). Commencing from the earliest consensus algorithm used in Bitcoin, proof of work (PoW) involves allocating billing rights and rewards according to the computing power. Other popular consensus algorithms include the proof of state (PoS) algorithm used in Ethereum, practical byzantine false tolerant (PBFT) algorithm and Raft used in Hyperledger Fabric. These consensuses are utilized for general single blockchains to guarantee security.

Considering multi-domain conversation systems, the main requirements include content reliability, flexible permissions, quick responses, and scalability. Regarding these demands (the limitations of existing consensus performance), a hybrid consensus mechanism for master–slave chains in multi-domain conversation systems is proposed herein.

5.2 Related work

As we discussed in Chapter 3, Hyperledger Fabric will be selected as the best fitting blockchain platform for our conversation system. In this section, we will review the exsiting consensus algorithm in Hyperledger Fabric and analyze the performances and insufficiency of these algorithms.

5.2.1 Existing consensus algorithms in Hyperledger Fabric

Kafka, Raft and PBFT are most popular consensus algorithms used in Hyperledger Fabric.

Kafka: is a distributed consensus algorithm that can manage messages in an ordering manner and ensure data consistency among multiple redundant copies. The Kafka orderer service obtains the data with corresponding topic from the Kafka cluster associated with Zookeeper to ensure the order of transaction data. The ordering service client can be connected to multiple OSN(ordering service nodes), and the OSNs do not communicate directly.(Kreps et al. 2011)

Raft:is a distributed crash fault-tolerant consensus algorithm, which can ensure that the system can still process client requests even if some nodes in the system have non-Byzantine faults. Technically speaking, Raft is a consensus algorithm for managing replicated logs, which is part of the replicated state machine.(Ongaro & Ousterhout 2015)

PBFT: is based on communication between different nodes and they are mostly used in private chains having authenticated nodes. PBFT uses permissive voting, with the principle that the minority is subordinate to the majority. PBFT is using typical three-phase protocol including pre-prepare, prepare and commit.

Kafka and Raft are distributed crash fault-tolerant consensus algorithms, which emphasize serialization and append only rules, But most of them do not consider Byzantine fault tolerance, that is, they only consider non-human issues such as system node failures and network failures, and do not consider malicious nodes to tamper with data (Su & Huang 2012). While PBFT is is widely used in distributed systems, however, classic PBFT is still a C/S response mode rather than peer-to-peer communication, and the design of PBFT in Fabric was oriented to distributed system execution based on the state machine replication to ensure request in the sequence can be executed correctly in the distributed system. and the consensus nodes in PBFT is fixed that cannot cope with the dynamic changes of nodes, especially it cannot perceive increase of nodes. Actually with the increasing of nodes, and the fault tolerance of the

system should be enhanced, but the original PBFT algorithm is still calculated according to the previous number, which is undoubtedly a waste of resources (Sukhwani et al. 2017).

5.2.2 Algorithm Optimization

According to the above bottleneck of existing consensus algorithms, the improvement of consensus algorithm that suitable for our proposed master-chain in conversation system should focused on the items:

- Considering "upgrade and downgrade" consensus node dynamically based on their performaces from many aspects, which can prevent the malicious nodes and improve the security and quality of conversation system.
- Dynamically perception of the node construction to maximize resource utilization.

5.3 Procedure overview

The core concept of our presented hybrid consensus algorithm is to ensure security and consistent collaboration among slave chains and master chain. For the proposed master–slave chain model, the master nodes as well as endorsement nodes, leader nodes, slave nodes and anchor nodes are dynamically updated. The consensus algorithm in slave chains is based on reputation-driven voting, while the consensus algorithm in the master chain is based on global and dynamic PBFT. To achieve this, three key modules are proposed, as follows:

Module 1—Consensus based on reputation-driven voting in each domain (R-V consensus): for each domain, the node with highest reputation will request vote and will be assigned as leader node when it grant more than half votes. For each type of transaction, the corresponding experts that satisfy the minimum reputation requirement will be selected as endorsement nodes to verify the received transaction and signature. If the transactions are legal and the signatures are correct, leader node will order transactions to a new block and broadcast to all slave nodes to make consensus. And within a predefined time frame, the leader node will send the summary of domain blocks to master chain to backup.

Module 2—Dynamic construction strategy for master nodes: The top 10% of the domain nodes in each organization, with the highest evaluation value in each slave chain, will be selected as master nodes. The evaluation value will be considered on the basis of its reputation, computing power, transaction activity, and times selected as the master node. Since the number of slave chain nodes and their reputation

values are both constantly updated, the master nodes will also be dynamically updated. The master nodes will collect the summaries of transaction blocks from leader nodes, package them into a verification block, and then approve and publish it. Thus, all the transaction blocks approved by the slave chains will be added to the corresponding summary into master chain. In this way, the transaction blocks will be consistent between the master chain and the slave chains.

Module 3—Incentive scheme: is one of the key elements of the decentralized conversation system that influences the behavior of participants by changing the relative costs and benefits of choices those participants may make. Incentives include both economic-based model and non-economic model that systems compensate individuals with token and reputation will be applied into our decentralized conversation system, and the proposed consensus algorithm is used to update their reputations and tokens.

In the following sections, we present the detailed processes of the hybrid consensus algorithm for the master–slave blockchain.

5.4 R-V consensus

In this section, we discuss the details of the R-V consensus, which is a consensus mechanism based on reputation-driven voting, which includes the selection of target consensus nodes, consensus processing and consensus confirmation.

5.4.1 Selection of target consensus nodes

For each transaction, we will select the target consensus nodes to construct the verification network, as shown in Figure 5.1. Target consensus nodes can be selected from a single domain or multiple domains, since the transaction may require cross-chain cooperation.

For single-domain transactions, the nodes that satisfied with minimum reputation in a single slave chain will be assigned as the target consensus nodes to construct the verification network. While for cross-domain transactions, each domain node requires inter-chain certifications approved by the master chain. If approved, they will be assigned to the list of target consensus nodes; otherwise, they will be moved into candidate node set. In this way, the target consensus nodes will be dynamically selected to construct our verification network.

The consensus nodes in our designed are divided into two levels: target consensus node(T) and

candidate consensus node(C), T is represented by set S_1, and C is represented by set S_2. T is the node that involve the concensus confirmation and get incentives. We set the maximum number of malicious node in T is f, then $|S_1| \geqslant 3f + 1$. C is the candidate node that is used to eliminate the list of consensus nodes at the end of the T after a period of time. We set $|S_2| = 2f$.

In the target consensus node set T, it has leader node l, endorsement nodes e, and other slave nodes s. The leader node in slave chain is selected as the node with the highest reputation in T. The endorsement nodes and other slave nodes are divided according to the radio of 2:3 based on their reputation values. The "upgrade and downgrade" of all the nodes is happened after a time period Δ.

5.4.2 Consensus Processing

To process transactions and reach agreements , a consensus mechanism based on reputation-driven voting is proposed to verify the transaction block, leader nodes append transaction-block summaries to the master chain after confirmation. Algorithm 1 presents a detailed description of the proposed consensus mechanism as the following four steps.

1) Transaction initiator will broadcast transactions to all target consensus nodes.

2) When the node receives the transaction, if it is not endorsement node, just flood forwarding; if it is endorsement node, will verify transaction, if legal, will signature with private key and send to the leader node; if it is illegal, discard it directly.

3) Leader node orders all the verified transactions to a new block and broadcast to all other target consensus nodes.

4) When the number of consensus that send ‘yea’ votes is greater than 2/3 of the number of target consensus nodes, the new block is valid and will be added to the corresponding slave chain.

5.4.3 Global PBFT Consensus Confirmation

The consensus confirmation is done by master nodes. The global PBFT consensus removes the first phase "Pre-prepare" from client. It is because our master chain is used to verify the inter-chain access authentication as well as backup domain blocks, which does not involve the sorting of requests and only focus on verification. Without client initiating a consensus, we will random select one node in master chain to initiate a consensus, because all the master node are endorsement nodes. Considering

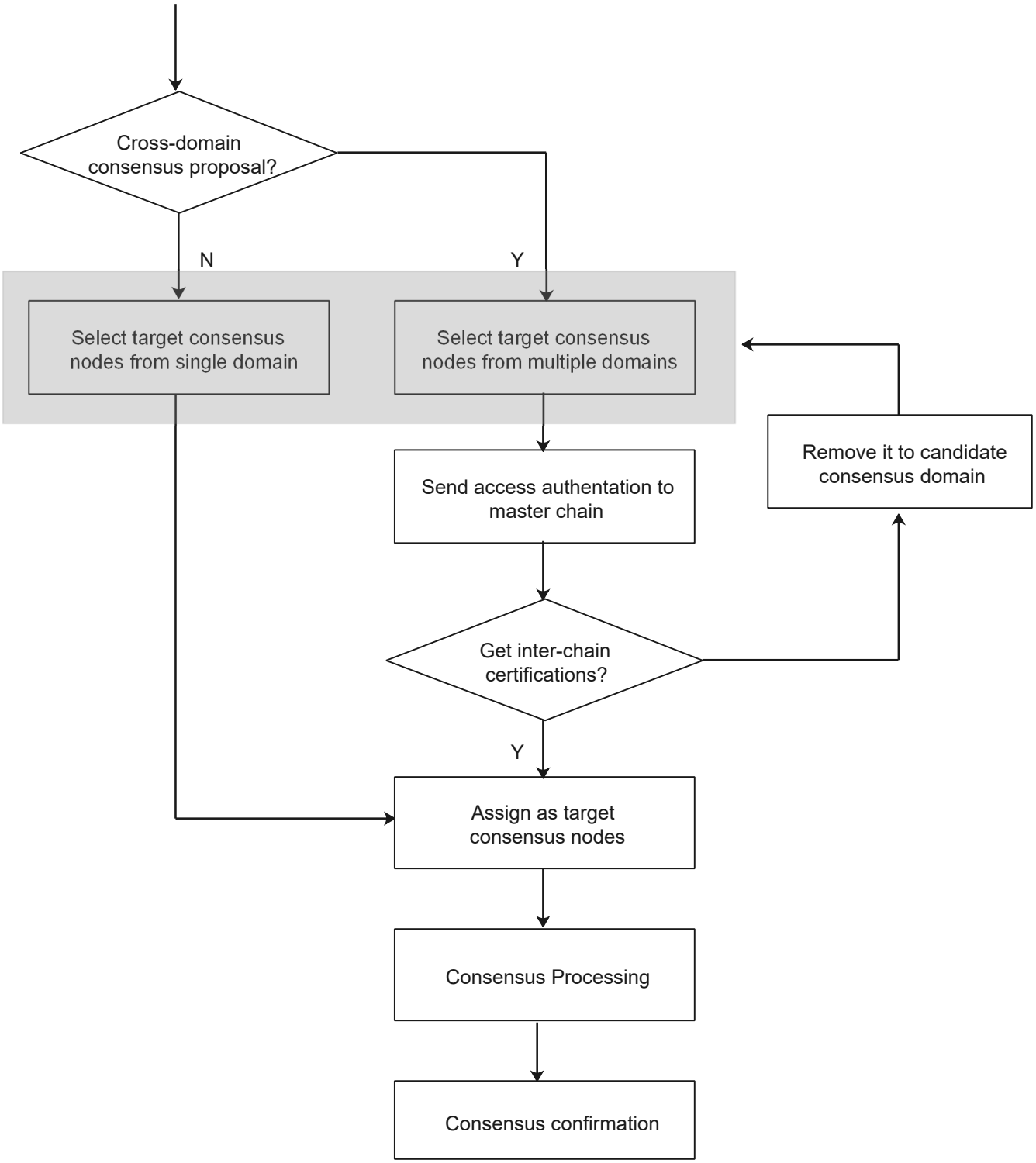


Figure 5.1: Selection rules for target consensus nodes.

Algorithm 1 Consensus based on reputation-driven voting

Input: transactions t,target consensus node set S_1, number of target consensus node A, candidate consensus node set S_2,leader node l, endorsement nodes e

Output: slave chain transaction block $Block_{sc}$, $blockSummary$

BEGIN

for each transaction t_i **do**
broadcast to target consensus nodes
if received node belongs to e **then**
verify the transaction
if transaction is legal **then**
Send it to leader node l
l orders the transactions and add into block $Block_{sc}$
broadcast $Block_{sc}$ to miners
if $\#preHash \leftarrow true$ or $\#currentHash \leftarrow true$ **then**
a vote send to all miners.
if received yea votes more than 2/3A **then**
index $\leftarrow int(Sha256(now \parallel trans[0]))$.
voucher $\leftarrow Sha256(now \parallel trans[index] \parallel index + 1)$.
add $Block_{sc_i}$ into slave chain i
create $blockSummary \leftarrow Sha256(Block_{sc_i})$.
else
discard illegal transactions
end if
end if
else
flood forwarding
end if
end if
end for

END

the simplicity and efficiency, We define the time interval to initiate a consensus as Δ. Thus, after a time period Δ, the leader nodes in each slave chain will send summary of domain blocks as well as inter-chain access authentication, then the selected master node will initiate a consensus with these transactions, and broadcast to other master nodes to verify and add into master block.

In the global PBFT consensus confirmation, since the master node will be dynamically updated, our fault tolerance of the master chain will be changed as well, thus it can perceive the number of increasing or decreasing of master nodes.

To sum up, in our proposed R-V consensus algorithm, if the domain node is satisfied with our minimum reputation value, it will be assigned as target consensus node to verify domain transactions. In this way, we can not only guarantee the verification and voting credibility, but also ensure that the transactions broadcasted to miners can be agreed upon and completed in a short time. Further, the block summary is created using the hash value of the transaction block, and will append a block summary to the master chain for backup. In this way, the data in the master chain and slave chains is consistent and tamper-proof.

5.5 Dynamic construction strategy for master nodes

In the proposed master–chain structure, the master nodes are constructed in real-time, based on their evaluation values. Our evaluation concept combines four factors: computing power (CP), transaction activity (TA), reputation value (RV), and times selected (T).

- CP indicates the computing infrastructure, which includes CPU, RAM, and bandwidth usage. Since it is related to operating and maintaining the blockchain network, it should be evaluated properly.
- TA indicates the degree of transaction activity of a domain node. Here, transactions in the conversation system are the created conversation rules. These transaction records can be considered to be an important indicator of participation activity for the entire conversation system.
- RV is the most significant aspect of our evaluation. It indicates the contribution-based reputation value of each node, which is used to represent the credibility of nodes.
- T indicates the number of times a node has been selected as a master node.

The evaluation value should be proportionate to CP, TA and RV, furthermore, to avoid the constructed right doesn't always belong to the previously selected nodes. We put T as a restriction factor to promote

circulation in the collection of master nodes. Eq (5.1) shows our calculation model.

$$EV = f\left(CP,\ TA, RV, T\right) = \frac{w_1 CP + w_2 TA + w_3 RV}{\sqrt{T}} \tag{5.1}$$

Where the coefficients $w_1,\ w_2, and\ w_3$ are the weighting coefficients of CP, TA, and RV, respectively. Their values range up to 1. Further,

$$CP = \alpha \log(PC_i E) \tag{5.2}$$

$$TA = \sum_{j=1}^{n} \log\left(T_j\right) A_j \tag{5.3}$$

$$RV = \frac{r_{current} + mean\left(\sum_{k=1}^{m} C_k\right) + r_{reward}}{3} \tag{5.4}$$

In Eq.(5.2), PC_i is the overall PC status we evaluated using the benchmark (Henning, 2006), and E denotes the number of times it is selected as the endorsement node.

In Eq.(5.3), n is the number of participating domains, T_j is the number of transactions created in domain j, and A_j is the weight for each related domain.

In Eq.(5.4), $r_{current}$ is the current reputation value of a node, C_k is the contribution percentage of a new transaction, and r_{reward} is the related incentive reward. Thus, when a node creates a new transaction, its reputation value will be updated according to the average of the newly assigned contribution and $r_{current}$ and r_{reward}.

According to the calculation model established above, we obtain the evaluation value list in each organization and choose the top 10% of the nodes in each organization to construct our master nodes. We set a time frame; the inter-chain authentications and intra-chain block summaries submitted within this time frame Δ will be agreed upon using the PBFT consensus algorithm by the constructed master nodes. Then, the next round of collection of master nodes is used to verify the subsequent time frame.

5.6 Incentive scheme

In this section, we consider how to encourage nodes to maintain high credibility. Incentive schemes will influence the behaviour of nodes by changing the relative costs and benefits of choices they make. Our presented incentive scheme includes an economic-based model and a non-economic model that compensate participants with both financial and non-financial rewards.

The economic-based model is based on an issuance mechanism (He et al. 2018), decided by the total number of tokens, allocation ratio (domain nodes and master nodes), and attenuation parameters. All these factors should satisfy Eq.(5.5). While the non-economic model is not related to any financial factors, for the conversation system, our non-economic model will be designed based on multiple linear regression (Neter et al. 1996). The design of the non-economic model should focus on how to provide the participants with more opportunities to obtain good content; if they can obtain more opportunities to become endorsement nodes or master nodes with higher reputation values and transaction activities, they will have a higher chance of maintaining conversations to obtain greater rewards.

The allocation of the system reward for each node is shown below:

$$rd_i = \frac{e_i}{\sum_{n \in CS} e_n} Y_{token} + \frac{r_i}{\sum_{n \in CS} r_n} Y_{other} \tag{5.5}$$

$$e_i = \begin{cases} x_{sc} * a_{sc} * b^0 + x_{sc} * a_{sc} * b^1 + x_{sc} * a_{sc} * b^2 + \cdots + x_{sc} * a_{sc} * b^n, \ where\ a_i \in \ domain\ node \\ x_{mc} * a_{mc} * b^0 + x_{mc} * a_{mc} * b^1 + x_{mc} * a_{mc} * b^2 + \cdots + x_{mc} * a_{mc} * b^n, \ where\ a_i \ \in \ master\ node \end{cases} \tag{5.6}$$

$$r_i = c_0 + c_1 RV + \delta \tag{5.7}$$

where Y_{token} is the total number of tokens, Y_{other} is the incentive degree of non-economic parameters, and m is the total number of nodes in the entire conversation system (CS).

To calculate the economic reward e_i of each node, in Eq.(5.6), x_{sc} and x_{mc} are the domain block and master block intervals for reward attenuation ($x_{sc} < x_{mc}$), respectively; a_{sc} and a_{mc} are the initial rewards of each block for a slave chain and the master chain, respectively ($a_{sc} < a_{mc}$); b is the attenuation coefficient; and n is the number of blocks created.

In Eq.(5.7), the non-economic reward r_i of each node will be calculated according to RV and TA. c_0 is a constant, c_1 is the partial regression coefficient, RV is the independent variable, and δ is the error term. The non-economic reward obtained, r_i, will be used to update the nodes' reputation value

according to Eq.(5.4).

5.7 Evaluation and Discussion

In this section, we will evaluate the performance of the proposed consensus algorithm, including transactions per second (TPS), delays, and fault tolerance in order to verify the applicability of the proposed consensus mechanism for master–slave chains.

5.7.1 Experimental Design

In this section, we deployed our master–slave chain on the Hyperledger Fabric blockchain platform to evaluate the proposed consensus mechanism. In the multi-expert conversation system, the transactions must be submitted to experts (domain nodes), which may not give real-time replies. To perform our evaluation continuously and quickly, the transaction simulation module is used. A total of 75 simulated nodes in the three slave chains and one master chain are used, the initial reputation values of each node are set randomly from 0.7-1.0, and tokens with 100 FT (Fabric token) are taken for each node. The architecture of the designed system is shown in Figure 5.2

To ensure the consistency of the system and its release efficiency, we built a conversation system for blockchains using the Java language, based on a restful service architecture. The lightweight 'json' format was used to facilitate data exchange. The experiments were simulated on one computer; the specifications of the standalone server is same with mentioned in section 4.4

5.7.2 Simulated Datasets

The simulated transactions in the experiments are of two types; one is an inter-chain transaction with conversational rules and contribution assignment, another is the intra-chain transaction with cross-chain authentication. Each transaction is created using a registered blockchain account, signed with its private key. Table 5.1 and 5.2 give templates of two types of simulated transactions.

In total, we tested 5,000 transactions (4,000 intra-chain transactions and 1,000 inter-chain transactions), and set the size of each domain block to 512 kB; the size of the master block is 1 MB.

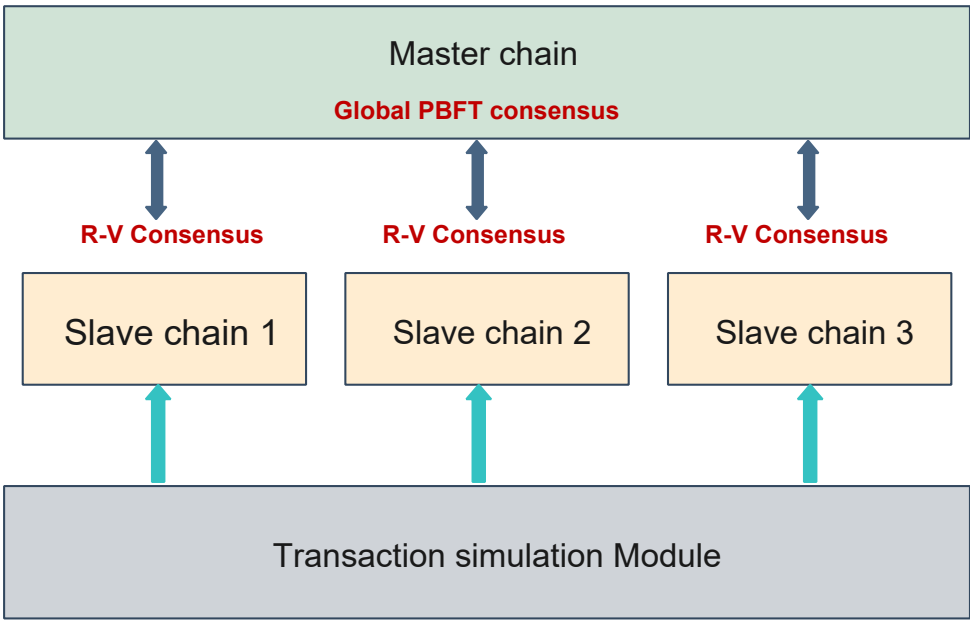


Figure 5.2: Structure of the designed system.

5.7.3 Performance Index

Three main performance indexes are used for evaluation and analysis in our experiments.

Transactions per second: This metric measures the time from when the transaction is broadcast to when transaction verification is completed, as follows:

$$TPS_{\Delta_t = SumTrans_{\Delta_t}/\Delta_t} \tag{5.8}$$

where Δ_t is the interval from transaction broadcasting to verification. $SumTrans$ is the total number of transactions during Δ_t.

Delay: This metric measures the entire time period from when the transaction is broadcast to when the block is broadcast, which includes the process of transaction verification and the process of block verification. This index is used to evaluate the network communication performance and consensus performance.

$$Delay_{tx} = TB_{tx} + TC + TB_{block} \tag{5.9}$$

where TB_{tx} is the preparation time when transactions are verified by endorsement nodes and broadcast to consensus nodes, TC is the execution time of the proposed consensus protocol, and TB_{block} is the verification time of each new block.

Fault tolerance: According to the proposed consensus algorithm, the maximum number of fault nodes

Table 5.1: Template of intra-chain transactions

Item	Rule	Contribution	Chaincode	Timestamp	Signature
Content	If/condition/ then /conclusion/	Expert1: xx% Expert2: xx% Expert3: xx% ...	XX	Time created	Owner signature

Table 5.2: Template of inter-chain transactions

Item	PeerList	Timestamp	Signature
Content	Chaincode Access list	Time created	Owner signature

in the slave chain $f_{sc} = \lfloor \frac{N_{sc}-1}{3} \rfloor$, while the maximum number of fault nodes in the master chain $f_{mc} = \lfloor \frac{N_{mc}-1}{3} \rfloor$. We will ascertain whether the transactions and blocks can be verified when setting different numbers of fault nodes in the slave chain and master chain.

5.7.4 Experiment regarding TPS and Delay

To verify the efficiency of the proposed consensus technique for master–slave chains, in our experiments, we assumed the experimental results when using classical PBFT as the ground truth for our master-slave chain. Further, to test our proposed consensus method for the master–slave chain model, the test network allocations are shown in Table 5.3.

From Table 5.3, it can be seen that we deployed three slave chains with two tests:

- Test one: the number of nodes in each slave chain is 20. Then, we chose the target consensus nodes that satisfied the minimum reputation value as consensus nodes. It is clear that the number of consensus nodes in each slave chain should satisfy the fault-tolerance rule: $N_{sc} = 3f + 1$, where f is 2, 3, and 3, which is the requirement to apply R-V consensus in slave chains. The master nodes are selected from the top 10% of the reputation values from each organization in slave chain, and

Table 5.3: Allocations in the test network

	Master chain (MC)	Slave chain (SC)		
		SC1	SC2	SC3
Test 1 with 67 nodes				
Number of nodes	7	20	20	20
Number of target consensus nodes	7	7	10	10
Test 2 with 100 nodes				
Number of nodes	9	30	30	30
Number of target consensus nodes	9	10	19	28

Figure 5.3: TPS performance results with different time interval of test 1 network.

the master chain utilizes the global PBFT consensus algorithm, the target consensus nodes in the master chain should be greater than $N_{mc} = 3f + 1$, so here N_{mc} is 7, where f is 2. The total number of nodes in the test 1 network is 67.

- Test two: the number of nodes in each slave chain is 30. Then, we chose the target consensus nodes that satisfied the minimum reputation value as consensus nodes. It is clear that the number of consensus nodes in each slave chain should satisfy the fault-tolerance rule: $N_{sc} = 3f + 1$, where f is 3, 6, and 9, which is the requirement to apply R-V consensus in slave chains. The master nodes are selected from the top 10% of the reputation values from each organization in slave chain, and the master chain utilizes the global PBFT consensus algorithm, the target consensus nodes in the master chain should be greater than $N_{mc} = 3f + 1$, so here N_{mc} is 10, where f is 3. The total number of nodes in the test 2 network is 100.

From these two tests, we can see that as the nodes in slave chains increasing, the number of target consensus nodes changed as well and the fault tolerance is enhanced on both slave chain and master chain.

For the TPS performance test, we set time internal Δ_t as 10s,20s,50s and 100s respectively, and for each time internal, we test 20 times and average it as final TPS for each time interval. The TPS experimental results of these two tests are shown in Figure 5.3 and Figure 5.4 respectively.

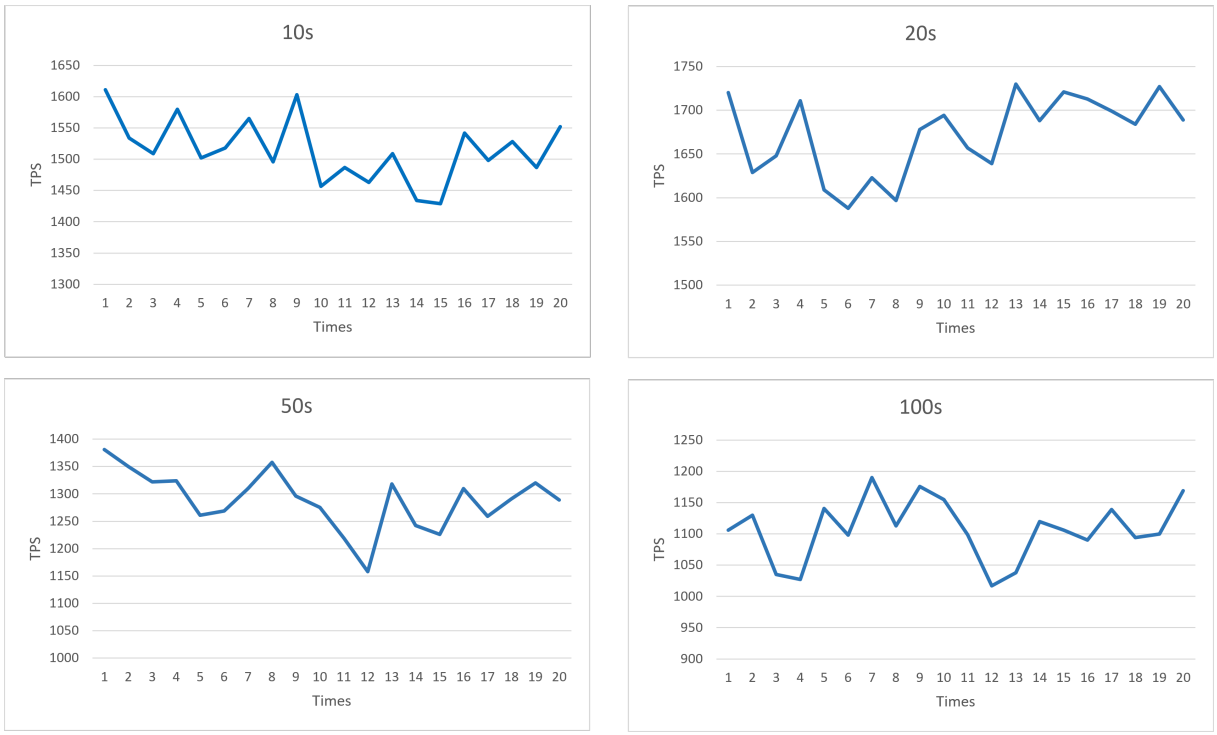


Figure 5.4: TPS performance results with different time interval of test 2 network.

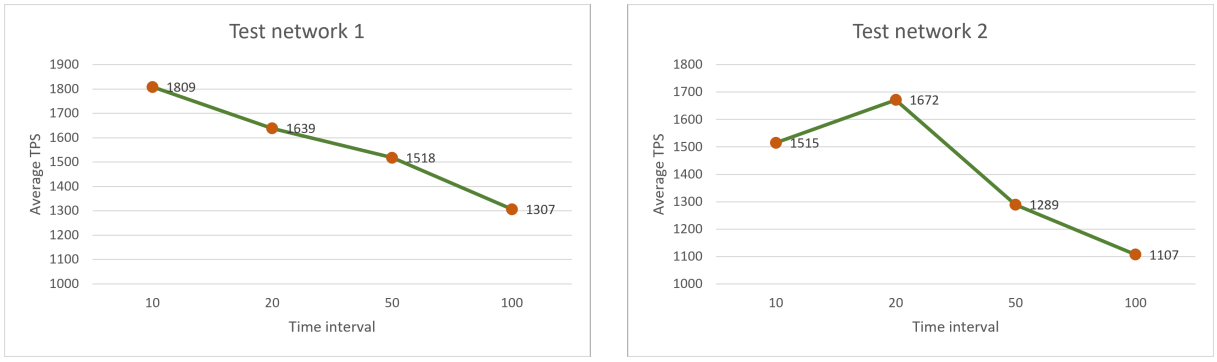


Figure 5.5: The relationship between average TPS and Time interval.

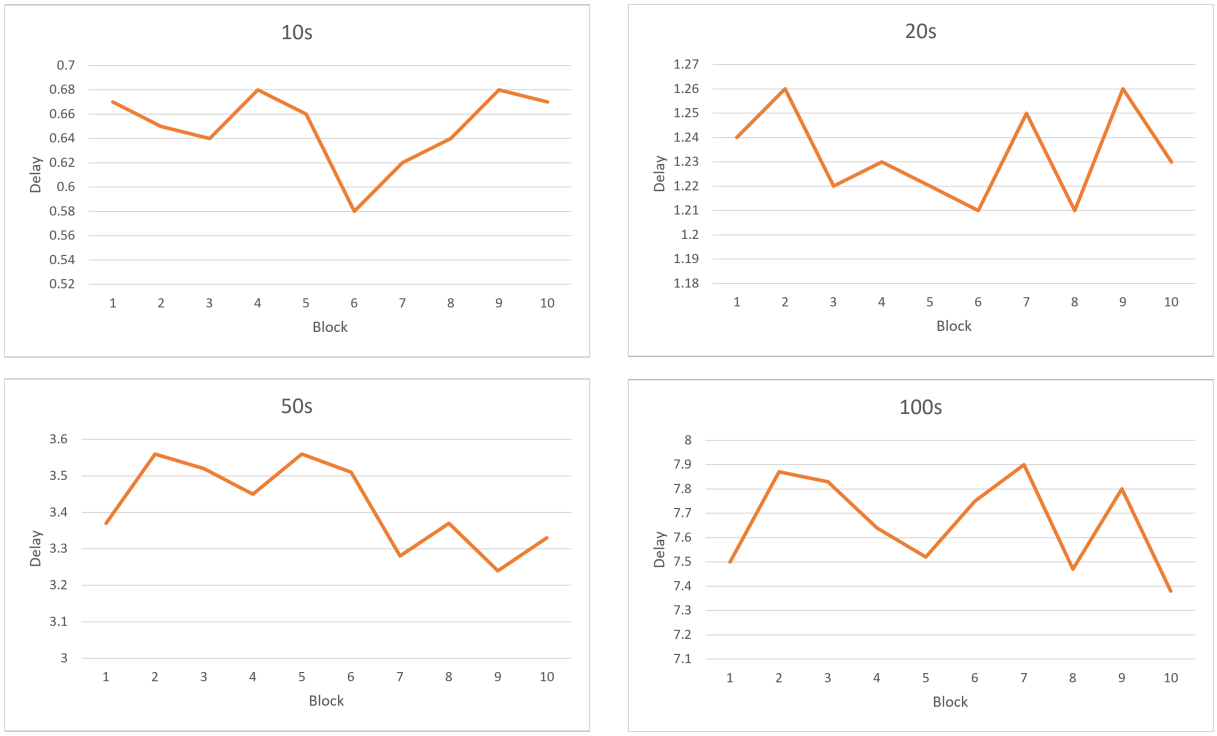


Figure 5.6: Delay performance results with different time interval of test 1 network.

Same with TPS experiments, we also set four different time intervals (10s, 20s,50s and 100s) to test delay and record the 10 block generation time. The Delay experimental results of these two tests are shown in Figure 5.6 and Figure 5.7 respectively. Figure 5.8 shows the relationship between time interval and Delay performance that the more delay along with the increasing of time interval.

As shown in Figure 5.5 and 5.8, different time interval will affect TPS and Delay. The longer time interval, target consensus node will receive more transactions during that time period and more transactions contained into block. And the bigger size of block will cause longer transmission and verification time that means longer delay. Thus, as the transactions increase, better TPS, however, if the transactions in the block exceed the processing capacity of target consensus node, the transactions will be accumulated and the processing thread will be blocked, then the TPS performance will drop down.

In the following comparison test, we set time interval in test network 1 as 10s and time interval in test network as 20s, then we compare the TPS and Delay performance between classical PBFT and our proposed consensus algorithm. Figure 5.9 and 5.10 shows the comparison results respectively.

As shown in Figure 5.9, with the same total number of nodes of 67 in test network one, the average number of verified transactions is about 167 per second in classical PBFT and 182 per second in our proposed consensus, while in the test network two, the average number of verified transactions is about 151 per second in classical PBFT and 168 per second in our proposed consensus. The experimental results

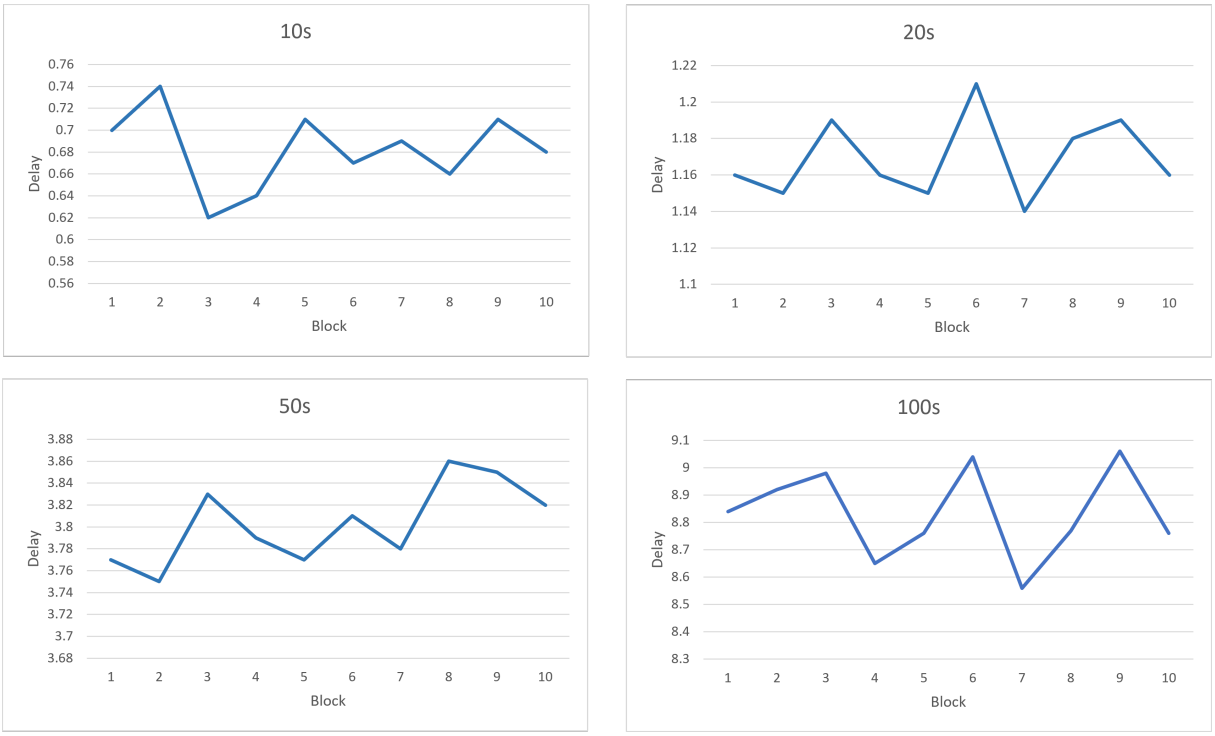


Figure 5.7: Delay performance results with different time interval of test 2 network.

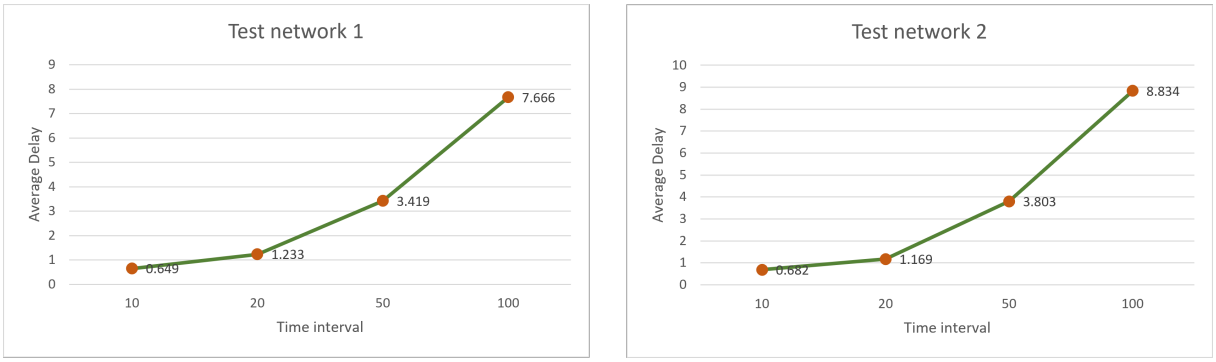


Figure 5.8: The relationship between average Delay and Time interval.

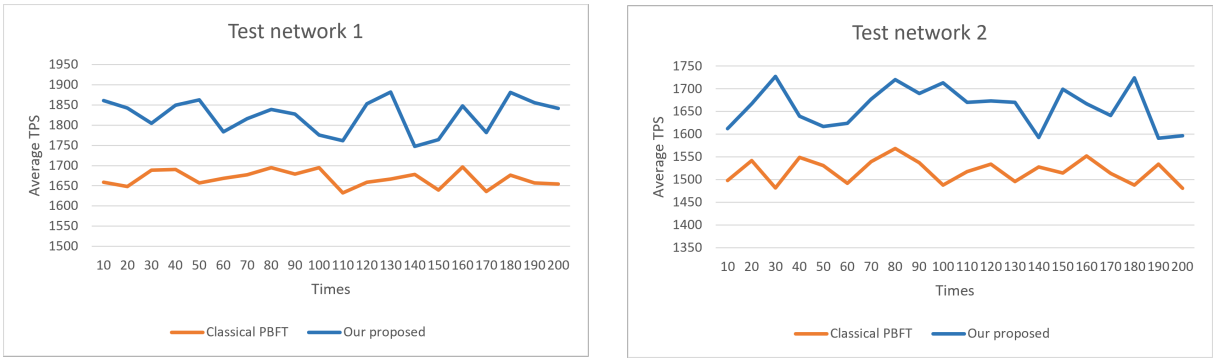


Figure 5.9: The TPS comparison results between classical PBFT and our proposed consensus algorithm.

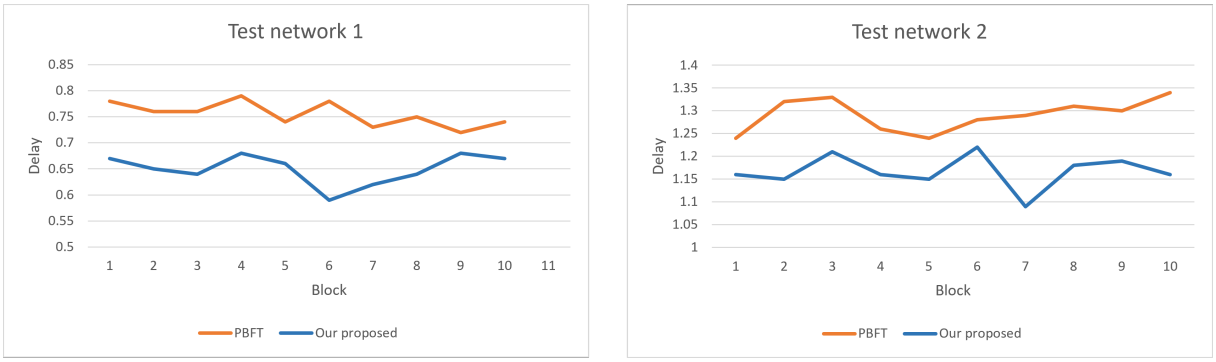


Figure 5.10: The Delay comparison results between classical PBFT and our proposed consensus algorithm.

for the average delay for 10 blocks are shown in Figure 5.10. It can be seen that the block generation time of our proposed algorithm is about 0.65s in test network 1 while the classical PBFT is about 0.76s, and in test network 2, the delay performance of our proposed algorithm is 1.17s and classical PBFT is about 1.29s.

We can see our proposed consensus improved the TPS performance as well as reduce the average delay compared with classical PBFT. One reason is because we define minimum requirement to set the target consensus nodes, which is an efficient way to reduce transmission and verification time. Another reason is that we remove the pre-prepare phase initiated by client, and set the time interval to initiate a consensus, which simplified three-phases PBFT to two phases.

5.7.5 Experiments regarding security

To test the security, liveness, and consistency of the proposed consensus algorithm, we verify the fault tolerance from three slave chains and one master chain. In the proposed R-V consensus in the slave chain, it is clear that the number of consensus nodes in each slave chain should satisfy the fault-tolerance rule: $N_{sc} = 3f + 1$. Since N_{sc} is 10, 19 and 28 in the three slave chains, then the f threshold will be 3, 6 and 9. Similarly, the number of master nodes is selected from the top 10% of the RVs from each organisation in the slave chain as the total number of 10, where the f threshold can be set to a maximum of 3. In our experiments, to set the node as faulty, we manually set its reputation below the minimum required RV value.

- Set different numbers of fault nodes in slave chain 1, $f_{sc} = 0, 1, 2, 3, 4$ and use TPS and delay to verify the feasibility.

- Set different numbers of fault nodes in slave chain 2, $f_{sc} = 0, 1, 2, 3, 4, 5, 6, 7$ and use TPS and

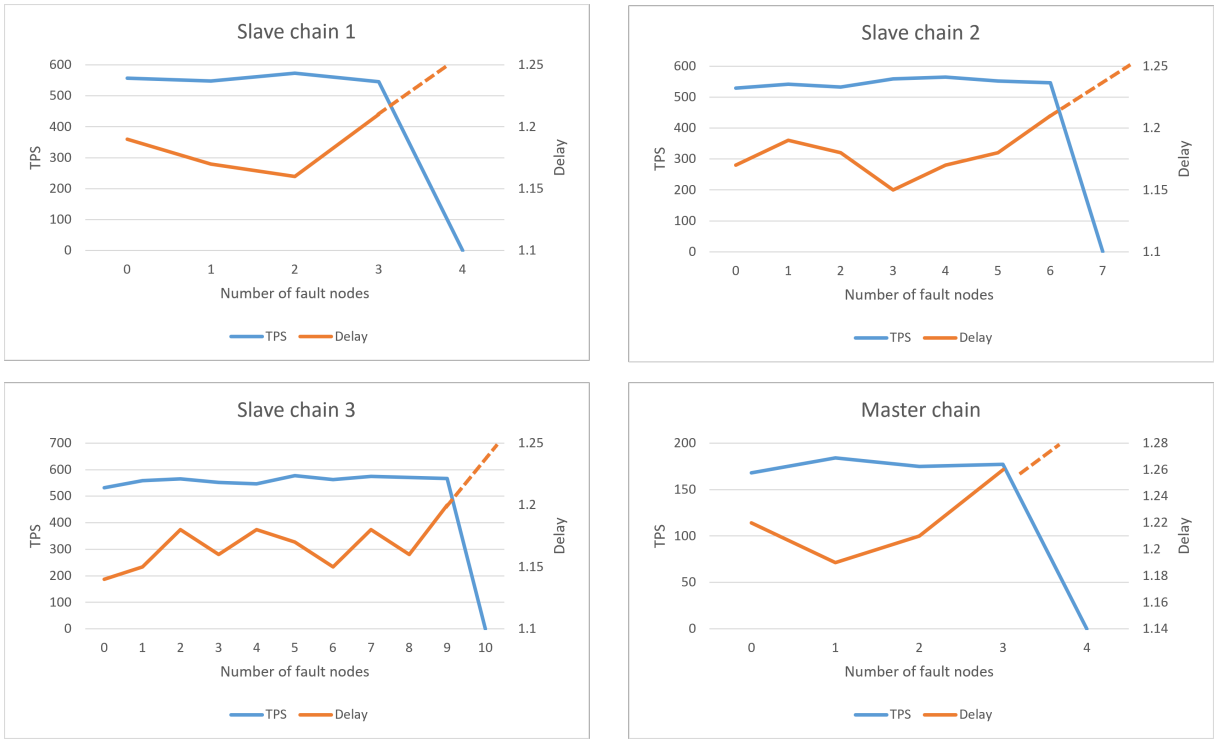


Figure 5.11: TPS and delay verification results with different numbers of fault nodes (three slave chains and master chain).

delay to verify the feasibility.

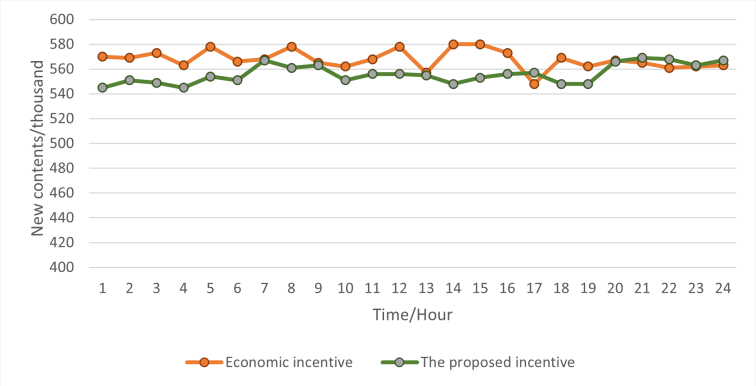
- Set different numbers of fault nodes in slave chain 3, $f_{sc} = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$ and use TPS and delay to verify the feasibility.

- Set different numbers of fault nodes in the master chain, $f_{mc} = 0, 1, 2, 3, 4$ and use TPS and delay to verify the feasibility.

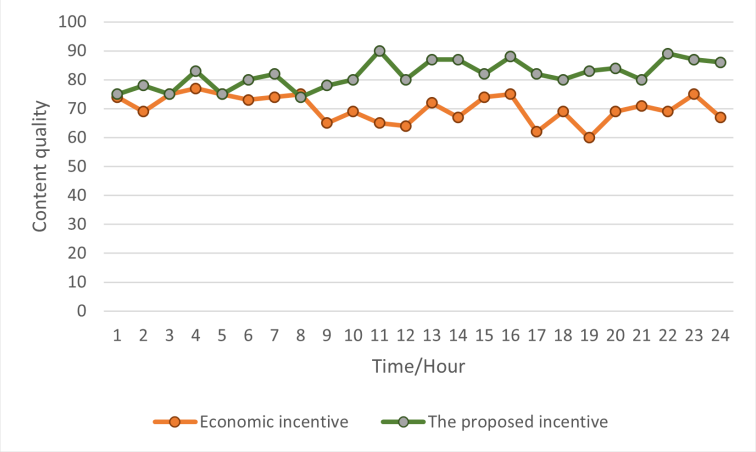
From Figure. 5.11, it is clear that if the number of fault nodes in the slave chain 1 is more than three, the inter-chain R-V consensus will not agree to add a malicious transaction or the related malicious block. Thus, the TPS=0 and Delay=$+\infty$ when f=4. Similarly, the TPS=0 and Delay=$+\infty$ when f=7 in slave chain 2 and f=10 in slave chain 3. Furthermore, if the number of fault nodes in the master is greater than three, the malicious transaction and block cannot be approved by global PBFT consensus.

5.7.6 Experiments regarding the proposed incentive scheme

In this section, I also evaluate the influence of the proposed incentive scheme compared with a traditional economic incentive model. Here, we set the total number of tokens as 300 FT per hour and investigate the number of new conversation contents, as well as the quality of new conversation contents by simulating



(a) The number of new conversation contents services



(b) The quality of conversation contents

Figure 5.12: Evaluation results between economic incentive and our proposed incentive

three randomised trials, and the duration of each simulation is 24 hours.

The results shown in Figure 5.12 were means of 3 tests. As shown in Figure 5.12(a), the amount of new content was a bit higher in the early-stage by using the traditional economic model, but the proposed incentive model can achieve a similar or higher number of new contents at the late-stage simulation. Overall, the traditional economic model and our proposed model perform similar in amount of new content created.

Furthermore, we also compare the content quality during this simulation. As shown in Figure 5.12(b), at the beginning of 8 hours, the economic incentive model and our proposed incentive model have similar influence of content quality. However, with the simulation time becoming longer, the quality of content is significantly greater when using our proposed incentive model. This is because the tokens were issued as a limited number and, with the increasing of participants, the influence of economic incentive will be decreased, and non-economic incentive will keep the impact no matter how many limited factors. By combining the economic and non-economic incentive models, we can see the

proposed incentive scheme will help the system to maintain quality, which is critical to build trustworthy for the conversation system.

In summary, we evaluated the feasibility and efficiency of the proposed consensus algorithm for master-slave chains. Based on the experimental results, it is clear that the proposed consensus method can greatly improve the TPS to support diversified conversation scenarios, and it also has the advantages of classical PBFT to maintain fault tolerance. In addition, the built-in incentive scheme, based on the proposed consensus algorithm, can provide stability and qualified conversation contents in the system.

5.7.7 Analysis of the experiments

Feasibility and scalability of master-slave blockchain scheme: the proposed master-slave chain provides a flexible chain structure that can enable concurrently process multi-domain transactions with high throughput. In the proposed scheme, multi-domain conversational interactions can be processed on the separated slave chains and then the generated blocks will be validated and confirmed by the master chain within the time internal, which can securely dispatch different conversation scenarios to different domain experts for useful blockchain scalability.

Efficiency of consensus algorithm: the proposed consensus algorithm for a master-slave chain improved the existing consensus algorithms used in Hyperledger Fabric, and is an effective and efficient mechanism that can improve the validation speed, as well as reducing the processing and transmission delay. This improvement occurs for two reasons: the first one is that we defined the minimum required RV value to set the target consensus nodes; this proved to be a dynamic way to reduce transmission and verification time. Another reason is that we removed the pre-prepare phase initiated by the client and set the time interval to initiate a consensus; this simplified the three phases of classical PBFT into two phases.

Security of consensus algorithm: with the proposed consensus algorithm for a master-slave chain, the system can guarantee security, stability and lively performance, regardless of the number of faulty nodes in the network. The experiment results regarding fault tolerance are also consistent with Proof 1 in section 4.4.1; these can always reach the correct consensus when no more than $\lfloor \frac{n-1}{3} \rfloor$ out of the total n replica nodes are faulty.

Stability and trustworthy of consensus algorithm: with the incentive scheme combined with economic scheme and non-economic scheme, the experimental results have shown that the built-in incentive

scheme based on the proposed consensus algorithm can provide the stable and qualified conversation contents in the system.

5.8 Summary

In this chapter, we designed a hybrid consensus mechanism comprising three modules: R-V consensus, construction strategy for the master nodes, and an incentive scheme. The R-V consensus is utilized for intra-chain verification, and the dynamic construction strategy is used to select master consensus nodes for inter-chain authentication. The incentive scheme is designed to generate both economic and non-economic rewards for all participating nodes during the consensus process. Finally, we evaluated the proposed mechanism from many aspects to verify its feasibility and efficiency to handle conversation interactions from different domains.

Chapter 6. Smart-contract enabled decentralized knowledge fusion for blockchain-based conversation system

Knowledge fusion used for handling cross-domain or complex questions in conversation systems has received considerable attention and interest. However, most existing knowledge fusion methods rely on centralized server, which face many limitations and challenges, such as a single point of failure, content tampering, and entrusted contribution assignment. In this chapter, we present a novel blockchain-based conversation system framework based on a decentralized knowledge fusion scheme using blockchain smart contracts to guarantee transparency, traceability, and non-tampering. Furthermore, we implement a system prototype based on our proposed master-chain structure and consensus algorithm in the Fabric network, the evaluation results of three case studies show the feasibility and effectiveness of the proposed decentralized knowledge fusion design in a conversation system.

6.1 Introduction

As previously discussed, to ensure the quality of interactions, it is better to use multiple experts rather than single expert to share and maintain conversational scenarios together (Nakano et al. 2011) (Nakano & Komatani 2020). Therefore, it is important to design an efficient and trustworthy mechanism to handle and maintain multiple conversation scenarios to support different services. A major constraint in managing the knowledge from multiple experts is the difficulty of cooperation among all experts and ensuring the knowledge understanding and representation. Most existing knowledge fusion methods can be used for global knowledge sharing and maintenance, however, the knowledge fusion methods used in conversation systems suffer from security issues.

To manage knowledge fusion from multiple experts, it is necessary to make appropriate fusion or global decisions and assign their contributions. The information from each expert should be kept

secure and identified clearly, so that, participants can share their knowledge as well as convince their contributions in a trusted and secure manner. Those information in the traditional mechanism is always stored in log files which can provide audit trails; nevertheless, these files are easily erased or alterable by unauthorized accesses to centralize servers. Therefore, it is difficult to merge knowledge from multiple experts using a centralized mechanism.

In addition, to support multiple conversation experts sharing and maintaining various conversation scenarios, we need to build an incentive scheme to encourage multiple experts to share good content and restrict bad content. However, the current incentive mechanisms are always determined by administrators or managers, which rely on certain authorities rather than a consensus for all participants and the results may not be trusted by participants. This is unfavourable for the sustainable development of our multi-expert conversation systems.

Furthermore, each expert's identification information and task solutions during conversation are always saved in the centralized sever, which has the darkest secrets of data privacy, and provide attackers with a single target to hack. Thus, the centralized control is not ideal for multi-expert conversation systems.

To address the above fundamental security issues in the current multi-expert conversation system, Blockchain has shown its potential for solving and enhancing traditional solutions with its key features: autonomous and decentralized processing, smart contractual enforcement of goals and traceable trustworthiness in tamper-proof transactions, and so on. Therefore, in this chapter, a novel blockchain-based decentralized knowledge fusion in a conversation system is proposed and implemented to guarantee conversation data security and provide reliable incentive scheme based on our proposed consensus mechanism.

6.2 Traditional knowledge fusion construction

In this section, we provide an overview of contemporary knowledge fusion construction, followed by their limitations, then implement the blockchain in multi-expert conversation system.

The knowledge fusion was proposed by Douglas, which was used in Cyc project focused on building a base of human consensus knowledge (Smirnov & Levashova 2019). The results from multiple experts may contain some redundant or wrong information, thus knowledge fusion is required to clean and integrate data.

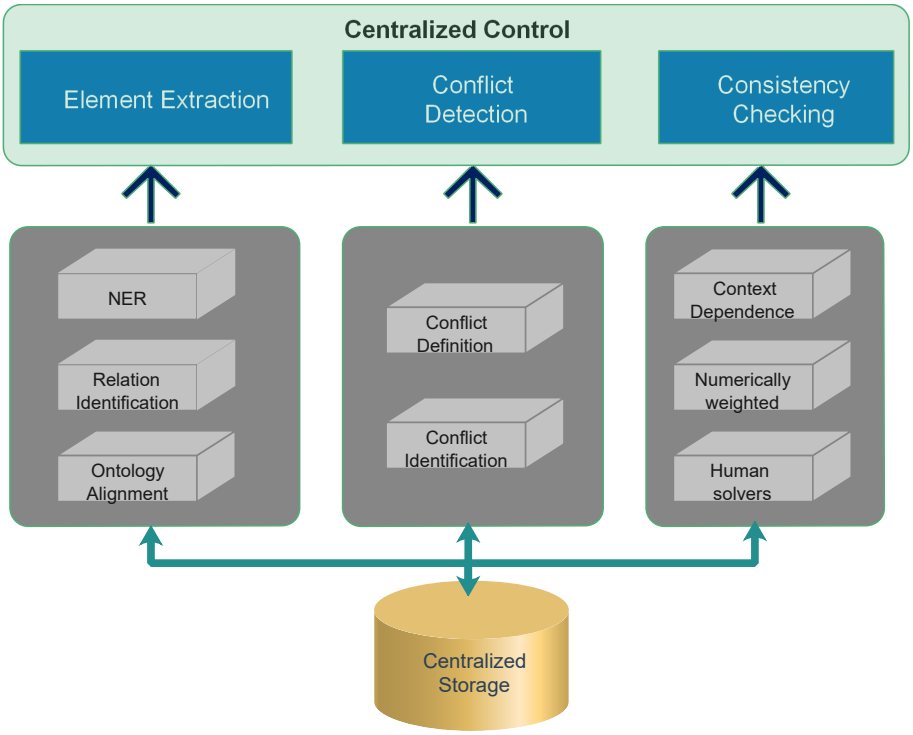


Figure 6.1: The traditional process of knowledge fusion.

As shown in Figure 6.1, the traditional process of knowledge fusion generally involves the following three components as below:

- Entity extraction: is the process of extracting corresponding entities and entity attributes between different experts' responses. The correspondences of elements can be matched as various relations, like equivalence, subsumption, disjointness or instance between entities of responses (Lian et al. 2017) (Zhao et al. 2018). The named entity recognition (NER), relation identification and ontology alignment are often used in this step.

- Conflict detection: if different responses use the opposite terms, predications, or semantic contexts, which are called conflicts (Pan et al. 2018) (Hertling & Paulheim 2020). There are many types of conflicts in natural language, including term conflict, predication conflict and semantic conflict. The rule-based models are utilized in this step.

- Consistency checking: this step will be focused on conflict resolution, which is needed where different responses have several conflicts and is used to facilitate a consistent knowledge (Zhao et al. 2016). Common techniques include numerically weighted constraint relaxation, context dependence and human problem solvers (Pradhan et al. 2017) (Ruta et al. 2018).

Obviously, traditional knowledge fusion is constructed in a centralized mechanism. It is of largest value gained for big corporations (such as Google and Amazon) to keep their knowledge bases maintained and organized collaboratively. However, such concentrated fashion has also created a growing number of limitations and challenges, e.g., contents tampering, unfair incentives and vulnerable to hacking. Thus, there is a critical demand of a new decentralized knowledge fusion for conversation system.

6.3 Blockchain-based conversation system framework

In this section, we present the overall blockchain-based conversation system framework and the general process for knowledge fusion for conversation system.

6.3.1 Overview Framework

The overview framework model of blockchain-based decentralized knowledge fusion in multi-expert system is shown in Figure.6.2. In the whole system, the decentralized framework has two main parts: unit chatting off-chain part and knowledge fusion on-chain part. The unit chatting part aims to fetch quick response from local knowledge base, while knowledge fusion part is utilized to finalize the fused response from multiple experts and assign contributions.

The workflow of the whole system is as follows: when user send a request, if the corresponding response can be found in the local knowledge base, the response will get back to user. Otherwise, the dynamic response will be finalized from multiple experts along with decentralized knowledge fusion and contributions will be added into blockchain database. Due to the features of blockchain, the knowledge fusion is launched via smart contracts and consensus algorithm is used to assign contributions as well as rewards and finalized response will be also updated in the local knowledge base for the next round.

6.3.2 The Decentralized Knowledge Fusion Process in conversation system

In this section, we describe the general process of our framework, our framework consists of six steps as follows:

Step1. Users send request to our conversation system, if the corresponding response can be found in the off-chain local knowledge base, the response will get back to user. If cannot, go to Step 2.

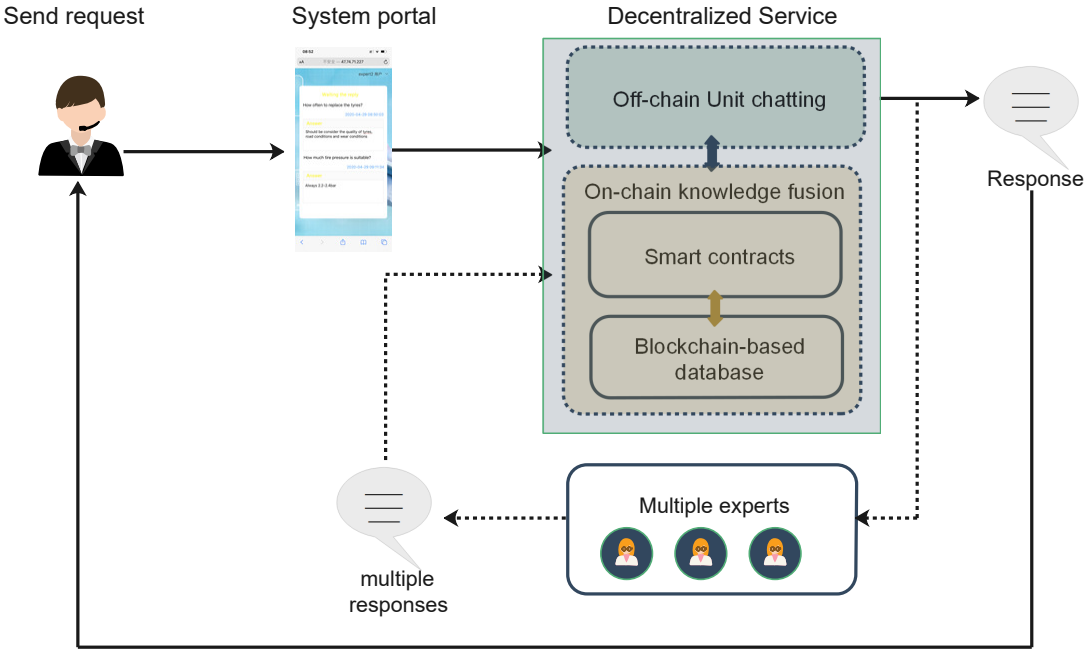


Figure 6.2: The blockchain-based conversation system framework.

Step2. The request will send to multiple experts who has been registered in conversation system. Each registered expert is assigned into a public key pair and their information will be written into a transaction stored in blockchain.

Step3. The multiple responses from multiple experts will be input to smart contracts which needs to be processed using on-chain knowledge fusion and validated by miners. The following steps are also related with this step and it depicts that the fusion solutions and contribution assignments are all recorded on the blockchain database permanently.

Step4. Registered experts receive the requests by interacting with system users. Each expert receives a request should deposit some tokens as well as reputation values to make sure the quality of the conversation contents.

Step5. Registered experts submit responses before due time. The expert with highest reputation will be add final fusion response and contributions for each participant into blockchain to request a consensus. After that, the final fusion result and corresponding request will be updated into local knowledge base as well.

Step6. Incentives including reputation and token are automatically assigned to experts according to the contribution assignment results. More contributions will get more tokens and improve the reputation.

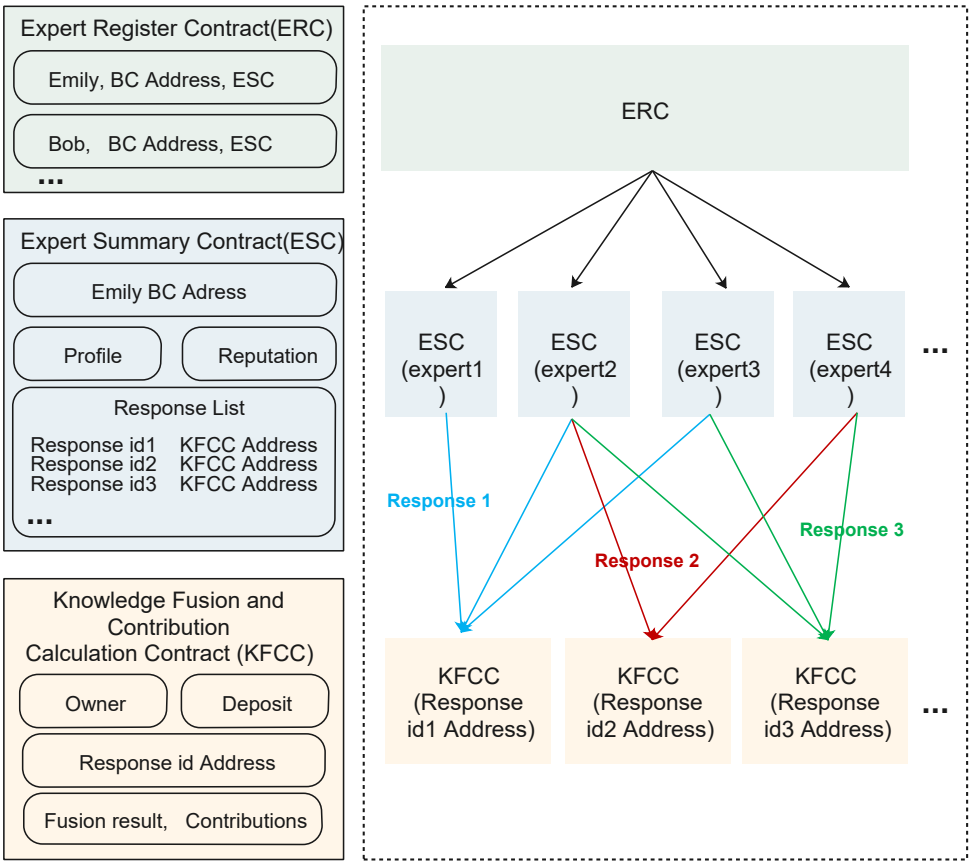


Figure 6.3: The blockchain-based framework for multi-expert conversation system.

6.4 Decentralized Knowledge Fusion Scheme

In this section, we present a concrete scheme of decentralized knowledge fusion for blockchain-based conversation system. The blockchain smart contract is adopted in our design. The whole process of knowledge fusion is written into blockchain smart contracts, which can be automatically executed in a trust manner. Based on the designed smart contracts, the formalized protocol is proposed to construct our system.

6.4.1 Smart Contract Enabled Knowledge Fusion

As mentioned before, the decentralized knowledge fusion will be used to build the response from multiple experts as well as assign contributions. In this work, we present three functional contracts for the implementation of decentralized service: expert register contract, expert summary contract, knowledge fusion and contribution calculation contract. Figure. 6.3 shows the contract structures and relationships.

6.4.1.1 Expert Register Contract (ERC)

The ERC corresponding to the new registered expert will be created. The expert registration contract is used to produce a unique blockchain *address* with a key pair (public key and private key) for the new registered expert. The address does not contain identity information about experts, which provides experts with privacy protection than experts in traditional conversation system.

To update or create an ERC contract, it needs to deposit transaction fee, which is given to miners who validate and confirm the transactions and support persistent running. And expert needs to pay a stipulated amount of token as well as reputation that can be withdrawn later if expert does not have any malicious behaviours.

6.4.1.2 Expert Summary Contract (ESC)

ESC contract is utilized to store the expert *profile*, *reputation*, and *response list* according to their past behavior, *Profile* will contain a digital signature signed by a certificate authority. If experts register with true identities, they can authenticate identities and updated by their public keys. *Reputation* is an important parameter which is initialized with a default value and updated with the completion of the knowledge fusion. *Response List* refers to the summary information about response statistics.

The above information will be set up when users first register and can be updated after knowledge fusion and contribution assignment. *Reputation* and *response list* cannot be changed by any experts. Each *response* in the *response list* has a corresponding address which will point to knowledge fusion and contribution calculation contract.

6.4.1.3 Knowledge fusion and contribution calculation Contract (KFCC)

KFCC contract depicts how to fuse multiple responses to get final solution and assign contributions for each expert, which is about the process of response receiving, fusion processing, contribution calculation and reward assignment.

When ESC posting response, KFCC contains a validation function to check if expert's reputation and reliability value satisfy the minimum limited. In general, a minimum reputation value is set to avoid low level experts. If experts are satisfied with default value, they can receive a response and participant the knowledge fusion. Then the knowledge fusion function is used to get fusion solution and calculate

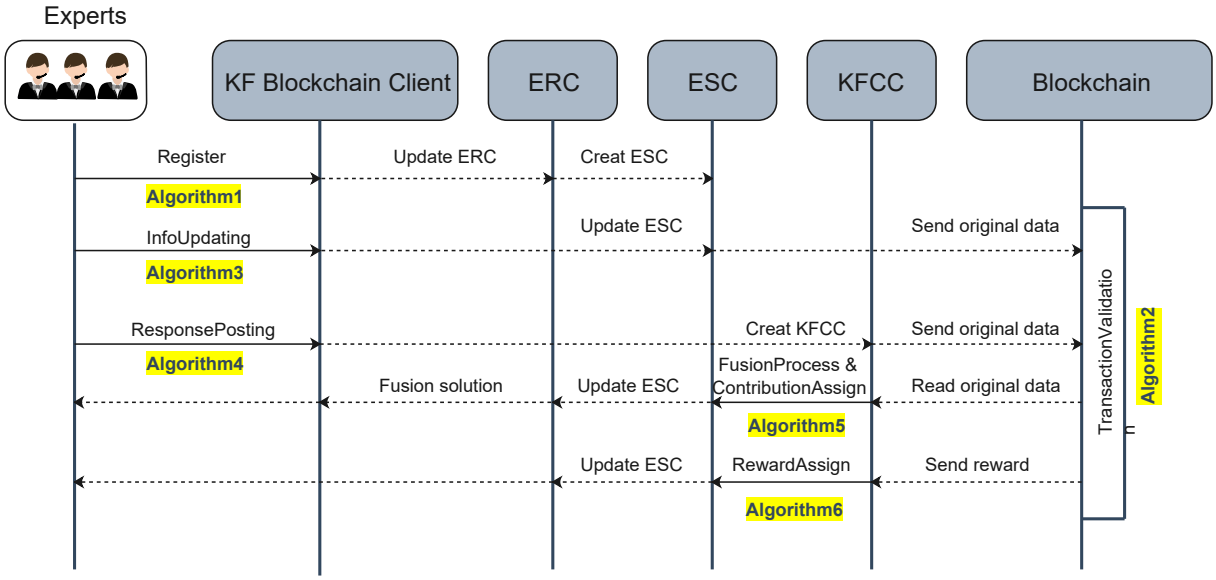


Figure 6.4: The general workflow of knowledge fusion in decentralized blockchain using smart contracts.

contributions. Different from the traditional method in which solutions are evaluated by centralized system, the fusion solutions as well as contributions in our system are reached consensus by miners. Meanwhile, all the contents are signed and stored with corresponding hash values into blockchain to guarantee unaltered at the source. Finally, reward assignment function is used to update tokens as well as reputation according to calculated contributions.

6.4.2 The Proposed Protocol

In the section, the concrete decentralized knowledge fusion protocols are designed to formalize our constructions. It consists of six algorithms: Register, TransactionValidation, InforUpdating, ResponsePosting, FusionProcess and ContributionAssign. Experts will interact with the blockchain by KF bockchain Client. The general workflow of decentralized knowledge fusion protocol is shown in Figure 6.4.

6.4.2.1 Register

In this algorithm, the experts will get their identities including blockchain address and a pair of key (public key and private key) via ERC contract, i.e.,$E_i = (B^a_{E_i}, B^p_{E_i}, B^t_{E_i})$. Meanwhile the corresponding ESC contract will be created, and the expert's initial reputation R_{E_i} and token T_{E_i} will be also set. Algorithm 2 illustrates the implementation of register process.

Algorithm 2 Register

Input: Expert E_i, expert initial reputation R_{E_i}, expert initial token T_{E_i}, ERC contract.
Output: $(B^a_{E_i}, B^p_{E_i}, B^t_{E_i})$ of E_i, update ERC contract, create ESC contract.
BEGIN

```
for each E_i do
    deposits the reward on blockchain
    if deposit failed then
        goto final
    else
        updateERC_Ei ← B^a_Ei, B^p_Ei, B^t_Ei,
        createESC_Ei ← R_Ei, B^a_Ei,
        withdraw deposited reward
        goto final
    end if
end for
```

END

6.4.2.2 TransactionValidation

The creating and updating of profile, fusion solution, contribution assignment can be both seen as transactions which needs to validate by blockchain miners and store into blockchain database. Experts who satisfy the minimum reputation can participate into blockchain as a target consensus node to make agreement and reach a trustworthy knowledge fusion process. To model the algorithm of transaction validation, we define the current and previous block as BC_c and BC_p respectively, and each block consists of blockchain height, previous hash of the block being checked, current hash of block being checked, timestamp, blockchain address of a miner, and transactions which needs to be confirmed. Algorithm 3 illustrates the implementation of transaction validation.

Algorithm 3 TransactionValidation

Input: Contract transaction T_{E_i} from expert E_i
Output: whether the validation of transaction is successful.
BEGIN

```
for each T_Ei do
    if #preHash ← true and #currentHash ← true then
        transaction verify via our proposed consensus algorithm
        Leader node blockchain address ← M_BC.
        index ← int(Sha256(now ‖ trans[0])).
        voucher ← Sha256(now ‖ trans[index] ‖ index + 1).
        create new block ← #blockheight + 1.
        output true as successful validation.
    else
        goto final
    end if
end for
```

END

6.4.2.3 InfoUpdating

After registration, experts can create and update their information into profile via ESC contract, and they can authenticate identities and updated by their public keys. The profile information of each expert should be validated as transactions, and if the validation is successful, the profile information will be updated into ESC contract. The implementation of information updating can be implemented as Algorithm 4.

Algorithm 4 InfoUpdating

Input: profile information $info_{E_i}$ from expert E_i
Output: Create and update ESC_{E_i}.
BEGIN
  for each $info_{E_i}$ **do**
    if E_i is unregistered. **then**
      E_i has not registered.
      $goto\ final$
    else
      CreateESUContract(ESC_{E_i});
      $Sign_{info_{E_i}} \leftarrow$ Digital signature on $info_{E_i}$ with $B^p_{E_i}$;
      TransactionValidation $\leftarrow$ $info_{E_i}$;
      if output true as successful validation. **then**
        UpdateESUContract $\leftarrow$ $ESC^{address}_{E_i}$,$info_{E_i}$;
      else
        $goto\ final$
      end if
    end if
  end for
END

6.4.2.4 ResponsePosting

After successful registration, experts can post responses Res_{E_i} to do fusion process and make contributions. In order to restrict bad contents, we specify that experts who post responses need to make a deposit including both reputation and tokens by $F_{E_i}(R_{reputation}, R_{token})$. Which can be withdrawn after response evaluation. And for each expert, we set the *responseEvaliation()* function to check whether the experts' reputations are satisfied with the minimum reputation condition R_{num} or not. If it is satisfied, the response will be updated to ESC contract and put into KFCC contract as well, otherwise will be discarded. Algorithm 5 illustrates the implementation of posting response.

Algorithm 5 ResponsePosting

Input: response Res_{E_i} from expert E_i ,deposit F_{E_i}, required minimum reputation value R_{min}, ESC address $ESC_{E_i}^{address}$.
Output: Create KFCC contract, update ESC_{E_i}.
BEGIN

```
for each info_{E_i} do
    if E_i is unregistered. then
        E_i has not registered.
        goto final
    else
        E_i deposit on the F_{E_i} blockchain.
        if deposits on blockchain falied. then
            goto final
        else
            responseEvaluation ← R_min, Res_{E_i};
            if E_i does not satisfy the minimum reputation condition. then
                goto final
            else
                Sign_{Res_{E_i}} ← Digital signature on Res_{E_i} with B^p_{E_i};
                UpdateESUContract ← ESC^{address}_{E_i}, Res_{E_i};
                CreateKFCCContract(KFCC_{E_i})
                Res_{E_i} put into KFCC_{E_i}
                withdraw deposited reward.
                goto final
            end if
        end if
    end if
end for
END
```

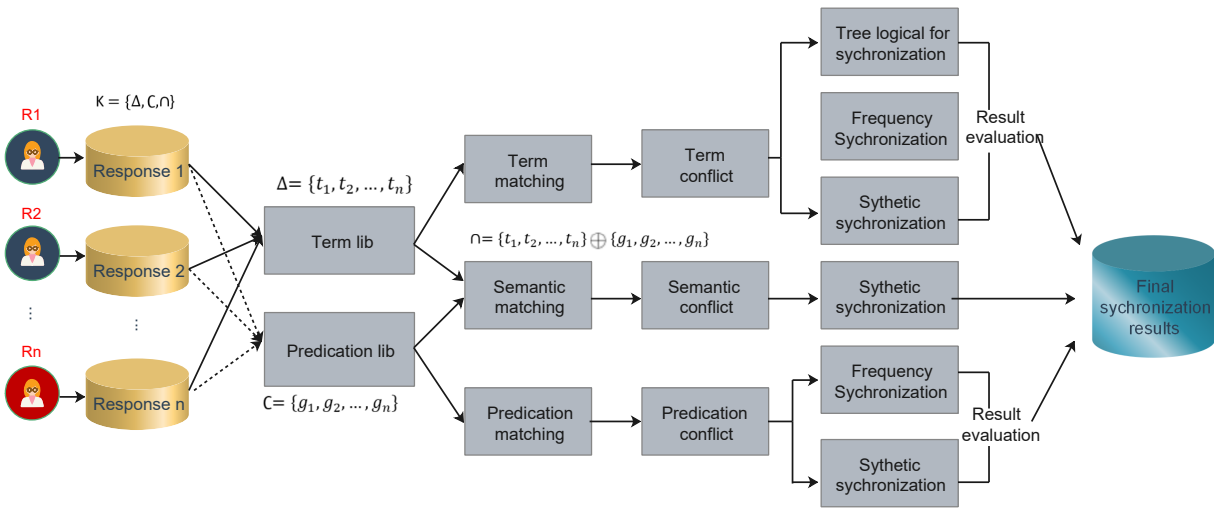


Figure 6.5: The main workflow if knowledge fusion process.

6.4.2.5 FusionProcess

All the evaluated responses will send to KFCC contract to process and obtain the final fusion result, the workflow of knowledge fusion is shown as Figure 6.5. It consists of three main steps:

Step1: Extract noun phrase lib and verb phrase lib: for each response, we extract the key noun phrases and verbs into term lib $t = \{t_1, t_2...t_n\}$ and predication lib $v = \{v_1, v_2...v_n\}$ respectively. Then we construct semantics lib $s = \{s_1, s_2...s_n\}$.

Step 2: Ontology matching: this step is based on WordNet (Leacock et al. 1998) consisting of a set of synonyms among synset, which denotes a concept of a group of nouns, verbs, adjectives, adverbs. A hybrid method based on WordNets (Jiang & Conrath 1997) is used to calculating similarity between noun phrases, verbs and semantics as below:

$$LS\left(c_i, p\right) = -\log\left(P\left(c_i \mid p\right)\right) \ = IC\left(c_i\right) - IC(p) \tag{6.1}$$

where the link strength (LS) is the difference of the information content between a child concept c_i and its parent concept p. Each concept consists of a set of term, verb and corresponding semantics $\{t, v, s\}$. $IC\left(c_i\right)$ is the information content of a child concept c_i while $IC\left(p\right)$ is the information content of a child concept p. $P\left(c_i \mid p\right)$ is the probability of child concept c_i linked from parent concept p.

The similarity between terms, verbs and semantics can be quantified as $sim\left(t_1, t_2\right)$, $sim\left(v_1, v_2\right)$ and $sim\left(s_1, s_2\right)$.

$$\begin{cases} sim\left(t_1, t_2\right) = \left|LS\left(t_1, p\right) - LS\left(t_2, p\right)\right| \\ sim\left(v_1, v_2\right) = \left|LS\left(v_1, p\right) - LS\left(v_2, p\right)\right| \\ sim\left(s_1, s_2\right) = \left|LS\left(s_1, p\right) - LS\left(s_2, p\right)\right| \end{cases} \tag{6.2}$$

Step3: Conflict resolution: based on the above ontology matching results, we can identify the conflicts among terms, verbs, and semantics. To get the fusion result, the three following synchronization methods are combined to facilitate a consistent knowledge.

- Synthetic synchronization: logical add of noun phrases, verb phrases, and semantics to eliminate

redundant phrases, verb phrases and semantics.

$$\begin{cases} Syntax\left(\Delta\right) = Syntax\left(t_1, t_2, \ldots t_n\right) \\ Syntax\left(V\right) = Syntax\left(v_1, v_2, \ldots v_n\right) \\ Syntax\left(\cap\right) = Syntax\left(s_1, s_2, \ldots s_n\right) \end{cases} \tag{6.3}$$

- Tree logical for synchronization: $logic(\Delta,\ T, N)$ where, Δ=($t_1,\ t_2$,... t_n), T is the logic tree of whole Δ, and N is the node of tree T. It is used to represent the relationship of phrases including kind belongs to concerning and inclusion relation.

- Frequency synchronization: $\mathrm{Frequency}\ (\Delta,\ \mathrm{f},\ \mathrm{X})\ ,\ \mathrm{where\ f}$ is the occurrence frequency of phrases or verb phrases. If $f_i > f_j$, then X=t_i, if $f_i < f_j$, then X=t_j, and if $f_i = f_j$, then X= $t_i\ or\ t_j$.

Based on the above steps, the fusion result will be given under the transaction validation and target consensus nodes on the blockchain should make consensus and confirmation. Algorithm 6 illustrates the implementation of fusion processing.

6.4.2.6 Contribution assignment

Contribution assignment is utilized to assign incentives including tokens as well as reputations. All the rewards will be confirmed by target consensus nodes and saved into blockchain. As shown in Algorithm 7, the contributions is measured via the text similarities between final fusion result and each expert's response. (i.e., high similarity will get more reward). The contribution assignment result will be automatically synchronized with expert's ESC contract to update their reputations as well.

6.4.3 Security Analysis

For our designed decentralized knowledge fusion, we used the proposed hybrid consensus algorithm to ensure non-tamperable guarantee. It fulfills the several security properties and we discuss as follows:

No Single Point of Failure: there is no centralized master server in the decentralized knowledge fusion model. If there are N ($N \geq 3$) target consensus nodes in knowledge fusion, to get the final fusion solution and assign contributions, more than 2N/3 miners should be reliable to reach agreement. Thus, our decentralized knowledge fusion model is exempted from no single point of failure.

No Third Party. Experts could share their response to get fusion solution and obtain reward

Algorithm 6 FusionProcess

Input: Evaluated response list $Res_{List} = \{Res_1, Res_2, ...Res_n\}$, response submitted time $T_{submit}^{Res_i}$, deadline $T_{deadline}$, KFCC contract.
Output: update KFCC contract and ESC contract.
BEGIN

```
for each Res_i in Res_List do
    if T_submit^{Res_i} ≤ T_deadline then
        nounPhraseLib △ ← extractNounPhrase(Res_i);
        verbLib V ← extractVerb(Res_i);
        semanticsLib ∩ ← extractSemantic(△,V);
    end if
    for each t_i in △, each v_i in V and each s_i in ∩ do
        matchedNounPhrase (t_i ↔ t_j) ← ontologyMatching(△);
        matchedVerb (v_i ↔ v_j) ← ontologyMatching(V);
        matchedSemantic (s_i ↔ s_j) ← ontologyMatching(∩);
        if matchedNounPhrases sets (t_i ↔ t_j) is not ∅; then
            relations ← treeLogicalSychronization(△);
            occurrenceFrequency ← frequencySchronization(△);
            nounPhraseFusionResults ← sytheticSchronization(△);
        end if
        if matchedVerb sets (v_i ↔ v_j) is not ∅ then
            occurrenceFrequency ← frequencySchronization(V);
            verbFusionResults ← sytheticSchronization(V);
        end if
        if matchedSemanticPhrases sets (v_i ↔ v_j) is not ∅ then
            semanticFusionResults ← sytheticSchronization(∩);
        end if
        output FusionSolution.
        Sign_FusionSolution ← Digital signature on FusionSolution with B_{E_i}^p;
        TransactionValidation ← FusionSolution;
        if output fusion solution as successful validation. then
            UpdateKFCCContract ← FusionSolution;
            UpdateESUContract ← ESC_{E_i}^{address}, FusionSolution;
        else
            goto final
        end if
    end for
end for
```

END

Algorithm 7 ContributionAssign

Input: Evaluated response list $Res_{List} = \{Res_1, Res_2, ...Res_n\}$, Final Fusion results $FusionSolution$, ESC contract.
Output: update ESC contract.
BEGIN

```
for each Res_i in Res_List do
    sim_i ← CalcalateSimilarity(Res_i, FusionSolution);
    rep_i^new ← UpdateReputation(R_i, sim_i);
    EndorsementNode E_endor ← Expert with highest Reputation(rep_i^new);
    E_endor create a block ;
    ContributionAssign add transaction into new block to assign contributions
of each expert ;
    Sign_ContributionAssign ← Digital signature on ContributionAssign with B^p_E_endor ;
    TransactionValidation ← ContributionAssign;
    if output fusion solution as successful validation. then
        UpdateESUContract ← ESC^address_E_i , rep_i^new ;
    else
        goto final
    end if
end for
```

END

without any third party. The fusion solution and contribution assignment records will be obtained through blockchain start contracts and stored into blockchain database, which means all the processing records and storage contents cannot be altered or deleted.

Trustworthy incentive scheme: reputation and token are two factors of reward for evaluating experts solving requests. In particularly, high reputation means high reliability to provide response and high probability to obtain tokens. In our decentralized knowledge fusion model, the reward assignment can only happen when expert really contributes a user request via KFCC contract, and ESC contract will be invoked by KFCC contract and updated after completing knowledge fusion process. In addition, all the smart contracts need to make deposit. Thus, if a malicious expert wants to change her/his reward assignment, she/he needs to tamper with a high cost as calculated in eq (6.2) and eq (6.3). In this way, we can expect that reward scheme by assigning reputation and token will work in a trustworthy manner.

6.5 Case Study

6.5.1 System design

We implemented the designed conversation system on Fabric blockchain to depict the knowledge fusion processing by blockchain smart contracts and test our proposed scheme. We evaluated the accuracy and

satisfactory by using pre-defined ground truth shown in Table D.1 in Appendices D. The overall system design is shown as Figure 6.7.

As shown in Figure 5.8 and mentioned in section 6.4, the fusion processing data and contribution assignments for a specific domain will be recorded and stored on the blockchain timely. In our case study, experts with minimum reputation value can submit their responses within prescribed time, then the multiple responses will be processed via smart contracts to create fusion record, which will be added into provider database as well as blockchain platform. All the records will be added via blockchain mining. And as a user node, they can send query request to retrieve records from blockchain and updated into local database.

The sample ledger shown as Figure 6.8. BC means whole blockchain, where B0 is the genesis block which only includes block header and block metadata, and the transactions will be included in the block data of following blocks. Each block Each block consists of three main parts: block header, block data and block metadata, and will be chained to previous block. The samples of metadata included in the master block and the domain block are shown in Figure 6.6. In the domain block, the metadata includes knowledge rules, the account information (blockchain address) of each participated expert and their contributions and chaincode. While in the master block, the metadata has peer list including the list of accounts belonging to different slave chains, the summary of the added domain block (that is the hash of block headers of domain block) and transaction hash.

For privacy and security reasons, the metadata will be encrypted before being inserted into the blockchain. The sample metadata of the domain block takes up about 350 bytes of information, while the information added into the master block is around 500 bytes. With further refinements we can minimise the number of rules in the domain block and the number of summary transactions in the master block, in order to improve the processing speed.

The world state database was implemented to address different types of conversation scenario. In our system, we created a set of 3 assets each with a unique identity: a different scenario, size, owner, and appraised value. The master chain is used to manage the summary of each blockchain domains as well as cross-chain authentication, while each blockchain domain has separated slave chains. Meanwhile, the new added knowledge fusion solution will be updated into local knowledge database via world state, which is easy to directly access the current value of a state rather than having to calculate it by traversing the entire transaction log (Venkatesh et al. 2018).

The designed system is deployed in the standalone server as one master chain and three slave chains,

```
{
 datetime: “2021-02-02 14:43:58”
 transaction_hash: “0xa2e2d044c711a932f248ecf1181fc0d68a685861da4031366b5c99eba1ecad83”
 metadata: [
 { “rule”: “if /condition1/ then /conclusion1/ ”,”contribution”: “account1: 0.84, account2: 0.77, account3: 0.90, account4: 0.92, account5: 0.85”,
  “chaincode”: 01 },
 { “rule”: “if /condition2/ then /conclusion2/ ”,”contribution”: “account1: 0.92, account2: 0.83, account3: 0.88, account4: 0.82, account5: 0.95”,
  “chaincode”: 02 },
 { “rule”: “if /condition3/ then /conclusion3/ ”,”contribution”: “account1: 0.89, account2: 0.78, account3: 0.81, account4: 0.90, account5: 0.98”,
  “chaincode”: 03 }
 ]
}
```


(a) Sample metadata in the domain block


```
{
 datetime: “2021-02-02 14:43:54”
 transaction_hash:
 “0x30b41fef9b68ff62e16231687199c6afff8834b30336cd8c357af9e1bce5601”
 metadata: [
 { “peer_list”: “chaincode1: account1, chaincode1: account2, chaincode2: account1, chaincode2: account2, chaincode3: account1”,
  “summary”: “transaction_hash: 0xa2e2d044c711a932f248ecf1181fc0d68a685861da4031366b5c99eba1ecad83,
   transaction_hash: 0x58a4083e349460e79d28d13da21bf3ab746186b8a5e897ab946f430cff9c0fc1” },
 { “peer_list”: “chaincode1: account2, chaincode1: account3, chaincode2: account1, chaincode3: account1, chaincode3: account2”,
  “summary”: “transaction_hash: 0xa05079b666dcec09214b4c8ae49a0b5ff499ca5a8c6791261ca986ba9d5a8525,
  transaction_hash: 0x66cfbd73933600b615d44ad9abdf4ee966a60fd6e9214a83155b001b176a7981” }
 ]
}
```


(b) Sample metadata in the master block

Figure 6.6: Sample metadata in the master-slave chain

and we simulate responses from five experts to test the our decentralized knowledge fusion scheme. Here we set the maximum prescribed time is 10 seconds, thus we will collect our the satisfied responses from multiple experts within 10 seconds and then send them to get fusion solution. To make sure each test request will have multiple responses, we also simulate and send out the each response from single expert randomly within 10 seconds when they received request.

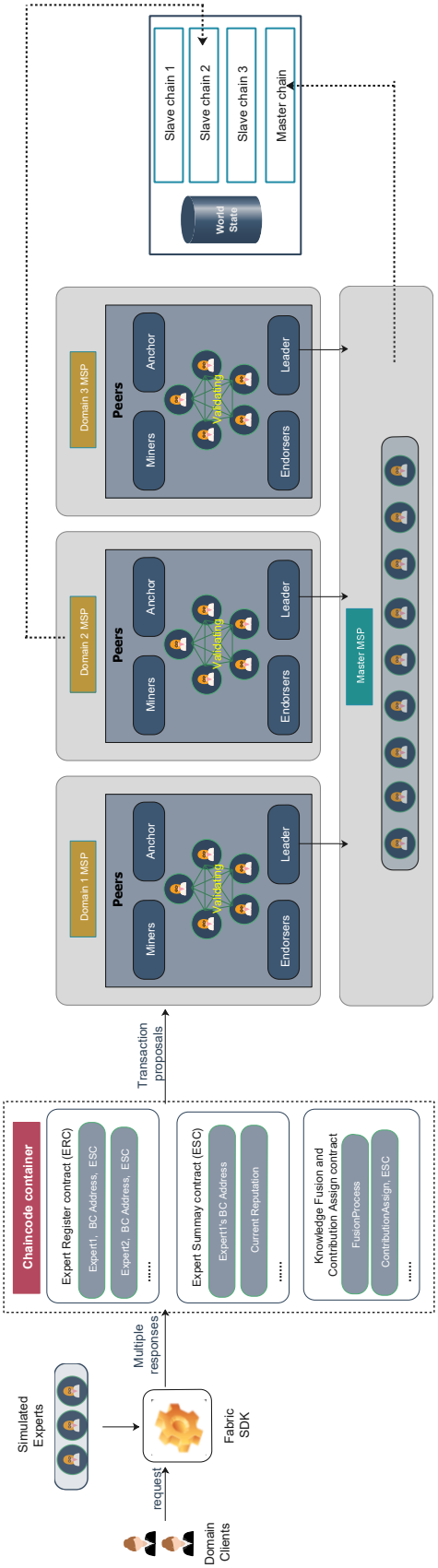


Figure 6.7: The Blockchain-based conversation system design of decentralized knowledge fusion.

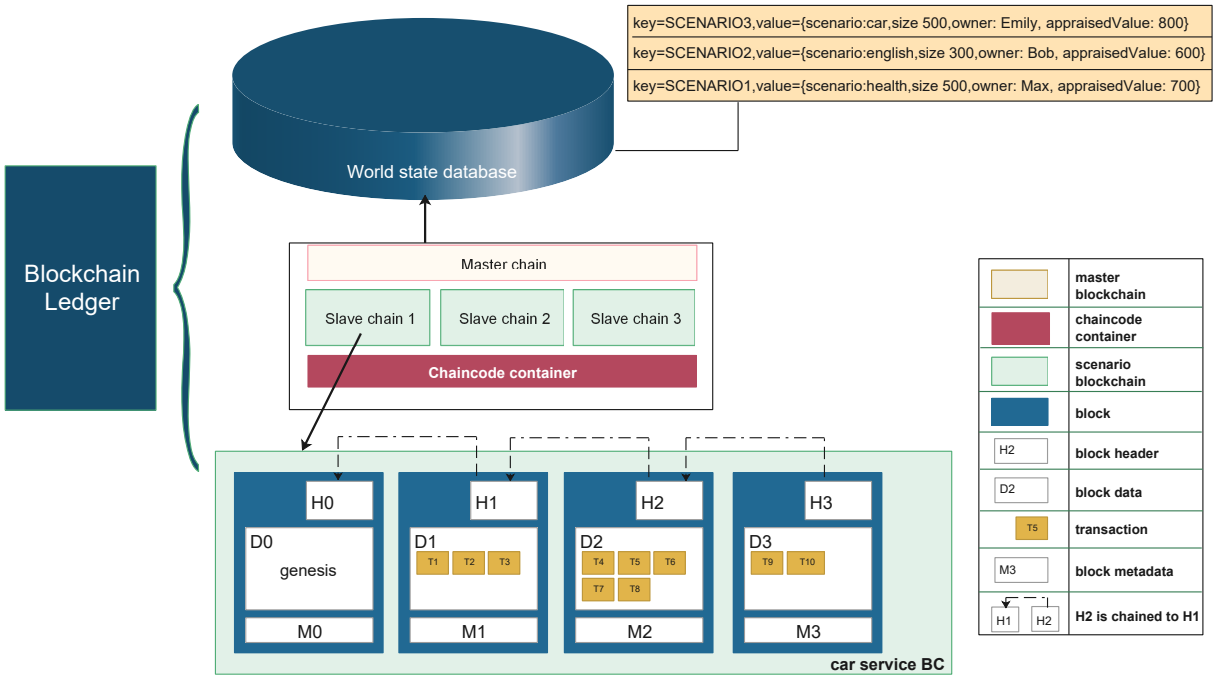


Figure 6.8: The sample ledger of the proposed system.

6.5.2 Case description

In order to verify the feasibility and effectiveness of the above conversation system based on decentralized knowledge fusion, we conduct three case studies related to car service, English learning and health &safety training. Specifically, for each expert, we identify whether this expert has enough reputation to process user request or not. Then through the decentralized knowledge fusion powered by blockchain smart contracts, we not only get a more reliable domain knowledge, but also record the processing tasks and making contributions via blockchain platform to make sure security storage and fair reward scheme. We used three main aspects to descript how to apply blockchain-based decentralized knowledge fusion in the conversation system: before, during and after knowledge fusion from multiple experts.

Before: each expert registered into our proposed system with a unique blockchain address. Experts have their base reputation, which used to check if it is satisfied with our minimum reputation requirement. Figure 9 shows the sample query of multiple experts' registered information in car service before decentralized knowledge fusion.

As shown in Figure 6.9, each registered expert will have a unique blockchain account, and we can query the current reputation from ESC contract as well as the blockchain account with highest reputation to let each participated expert know if he/she is the one with highest reputation or not.

During: when user asked one question, which can't find the matched conditions in the local

Figure 6.9: The sample query of expert registered information before decentralized knowledge fusion.

knowledge base (Hyeon et al. 2016), then this question will broadcast to registered experts. Then experts who satisfied with the minimum reputation can send their individual response. Multiple responses will be processed through our proposed decentralized knowledge fusion scheme and send the final fusion solution to the user. At the same time, based on the calculated contributions, the reputations will be also updated. The expert with highest reputation will be seen as leader node and claim a block by adding the knowledge fusion solution as well as calculated contributions into blockchain. Figure 6.10 show a sample conversation case to explain how it works by the proposed decentralized knowledge fusion:

- Domain client sends a request question “How often to replace tires?” to our system;
- There are five experts satisfied with our minimum reputation (>=0.80) to submit their responses within 1 min.
- For each response, the KFCC contract is utilized to get the final fusion solution and send back to user, which will be processed as mentioned in 6.4. For example, for the noun phrases fusion process, we extract and apply synthetic synchronization to get one term “degree of tire wear”, then use tree logical synchronization to identify the relationship with other terms, such as finding its synonym “quality of tire wear”, furthermore, calculating the frequency as 3 from all the responses. The order of extracted noun phrases as well as verb phrases will be decided by calculated frequencies. Then we apply Nature language generation(NLG) API to generate a natural language response base on identified synchronization information.
- Finally, we will calculate the text similarities among each individual response and final fusion

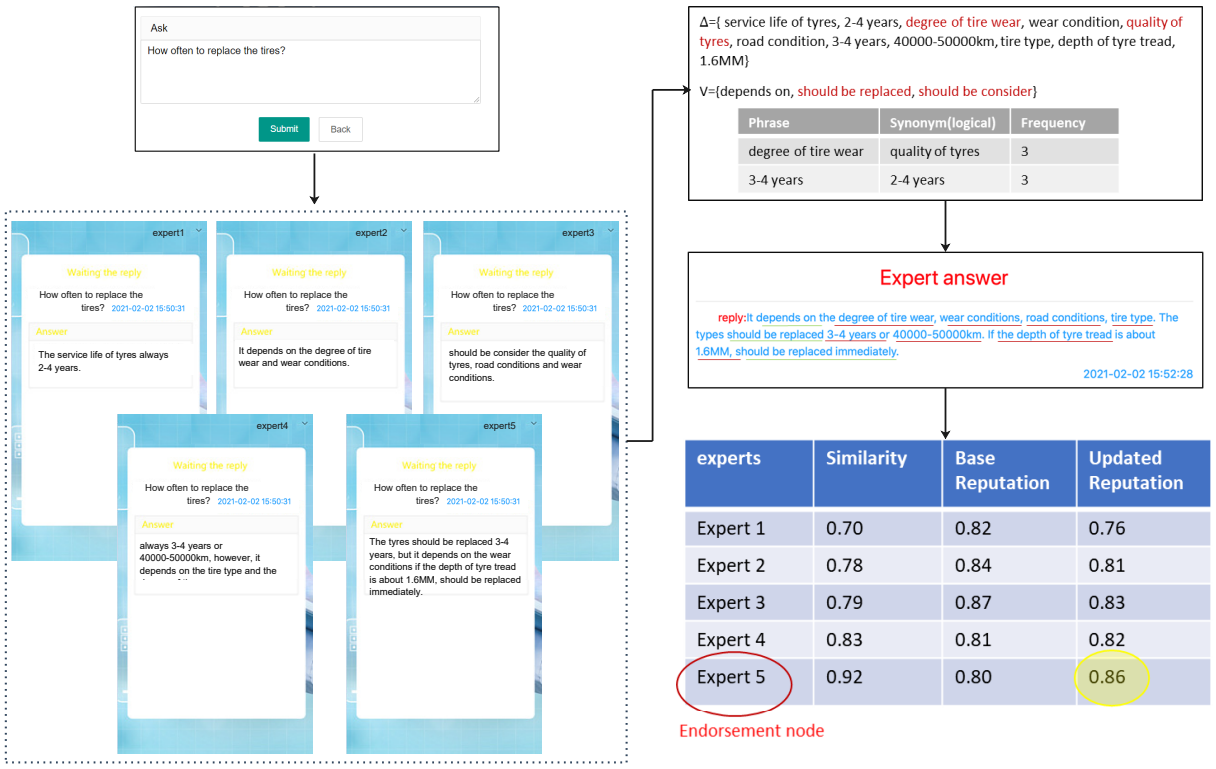


Figure 6.10: The sample conversation case during decentralized knowledge fusion.

solution to assign contributions and update the current reputations. For example, The expert 5 in this case will be the leader node.

After: each expert can query the updated reputation and token after knowledge fusion. The expert with highest reputation will add a new block with corresponding transaction into blockchain ledger and send the summary of block into master chain. Meanwhile, the new updated knowledge fusion as well as contributions will be updated into local knowledge base through world state.

As shown the sample case in Figure 4.5, Expert5 will add a new block into car service blockchain, and miners including all participants will be validated this block and stored into blockchain. Meanwhile, the request question and final fusion solution will be also updated into local knowledge base. In this way, it can not only secure the fusion process with non-tampered contents, but also provide a live maintenance of local knowledge base to make it keep updating with a trustworthy manner.

6.5.3 Evaluation and analysis

A total of 60 request questions are collected to evaluate our proposed decentralized knowledge fusion scheme in our designed conversation system, including 20 questions for car service, 30 questions for

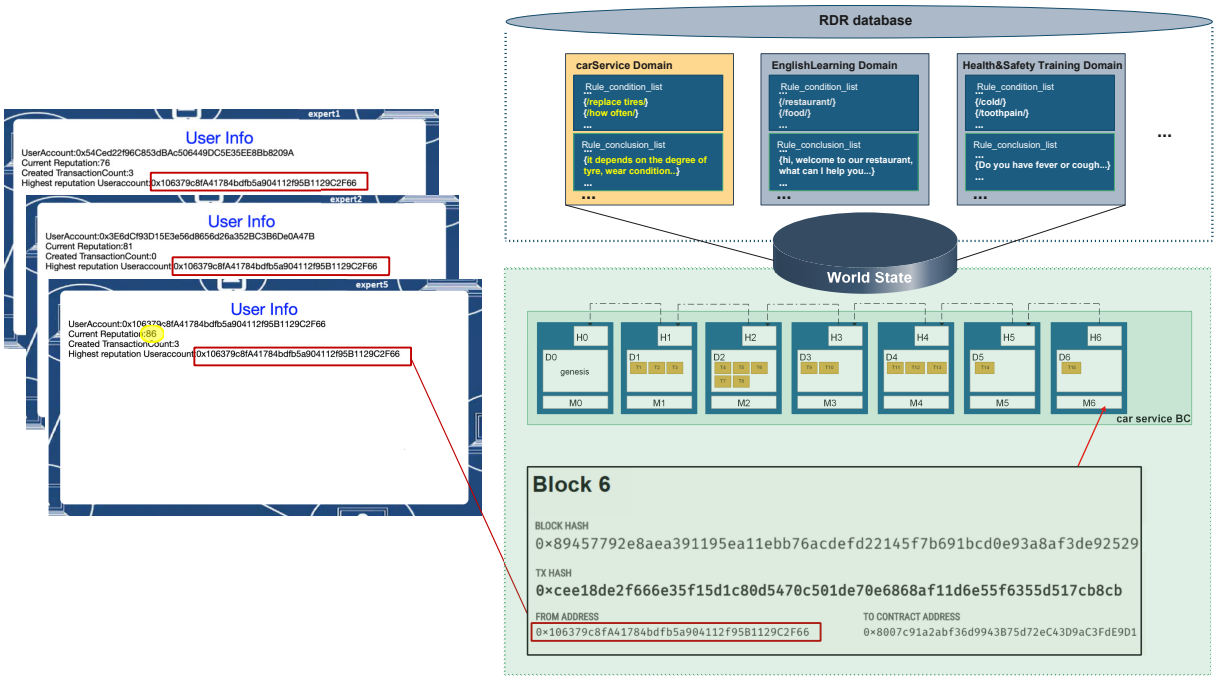


Figure 6.11: The sample block adding after decentralized knowledge fusion.

English learning and 10 questions for Health and Safety training. The ground truth regarding car service consultation is used from online car serving guide (n.d.), and the ground truth regarding English learning requests is used from (Ltd 2020), as well as the ground truth regarding health and safety training requests is used from UTAS (Stewart & Kokoris-Kogia 2020). For each case study, we simulate five experts to provide responses and perform our proposed decentralized knowledge fusion to evaluate the feasibility and accuracy. The simulated responses and fusion results of these three case studies are shown in in Appendices D.

In order to test fusion feasibility and accuracy, the semantic similarities used Twinword's tool (*Twinword* 2021) for evaluating whether the fusion responses are appropriate and useful for each conversation request (Zhou et al. 2020) (Zhang et al. 2020). Firstly, the semantic similarity between the fusion solution and the ground truth was measured, and the accuracy between a fusion solution and a single expert solution was also compared.Table 6.1 lists the summarizing results.

Table 6.1: The evaluation results between single expert and fusion results

ID	Contributions					Fusion
	Expert1	Expert2	Expert3	Expert4	Expert5	Semantic Similarity
1	46%	54%	84%	14%	54%	86%
2	27%	90%	99%	29%	80%	94%
3	21%	72%	83%	72%	21%	84%
4	78%	72%	91%	82%	96%	98%

Table 6.1 continued from previous page

ID	Contributions					Fusion
	Expert1	Expert2	Expert3	Expert4	Expert5	Semantic Similarity
5	64%	81%	64%	83%	82%	88%
6	77%	86%	75%	77%	73%	91%
7	92%	91%	82%	78%	74%	98%
8	96%	95%	92%	94%	96%	98%
9	78%	89%	77%	89%	90%	98%
10	58%	97%	89%	59%	15%	97%
11	44%	88%	44%	44%	57%	92%
12	13%	45%	88%	88%	60%	88%
13	37%	89%	94%	16%	56%	100%
14	87%	90%	98%	83%	83%	89%
15	82%	86%	86%	78%	90%	87%
16	49%	75%	84%	87%	77%	88%
17	82%	79%	86%	76%	80%	86%
18	68%	83%	90%	14%	86%	94%
19	34%	83%	91%	84%	84%	91%
20	81%	81%	76%	80%	18%	89%
21	59%	59%	59%	74%	51%	78%
22	71%	79%	78%	78%	37%	79%
23	70%	77%	70%	66%	70%	77%
24	71%	84%	70%	69%	71%	82%
25	70%	80%	73%	77%	34%	86%
26	57%	51%	72%	55%	55%	81%
27	75%	77%	75%	83%	80%	85%
28	80%	83%	80%	86%	83%	87%
29	63%	79%	64%	63%	81%	83%
30	67%	96%	97%	93%	63%	97%
31	73%	70%	74%	72%	70%	81%
32	80%	33%	75%	86%	86%	86%
33	86%	84%	92%	91%	85%	92%
34	96%	83%	96%	79%	83%	98%
35	83%	86%	74%	78%	68%	83%
36	44%	82%	81%	83%	74%	83%
37	68%	74%	83%	78%	74%	88%
38	90%	90%	90%	90%	79%	90%
39	56%	52%	100%	100%	82%	100%
40	84%	72%	77%	37%	84%	85%
41	79%	78%	83%	79%	49%	83%
42	33%	91%	29%	91%	91%	91%

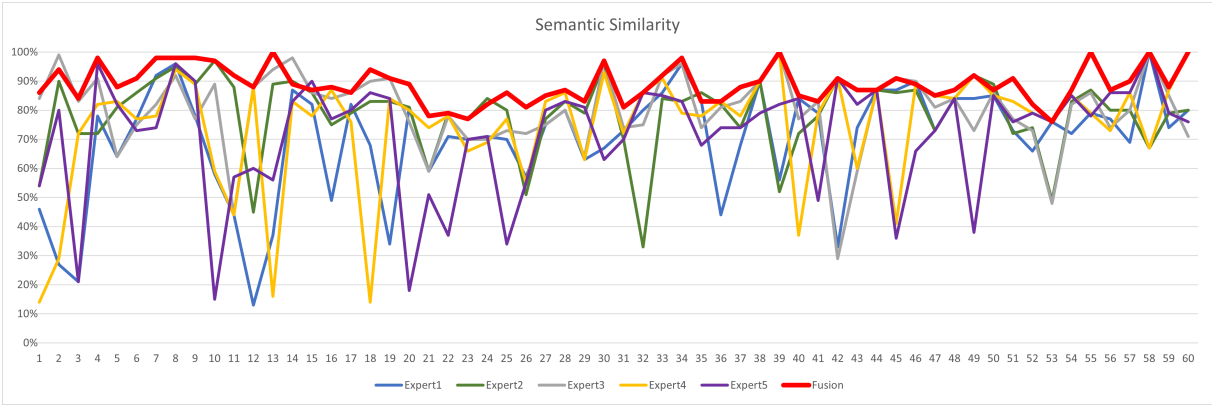


Figure 6.12: The comparison results between single expert and our proposed fusion scheme.

Table 6.1 continued from previous page

ID	Contributions					Fusion
	Expert1	Expert2	Expert3	Expert4	Expert5	Semantic Similarity
43	74%	87%	59%	60%	82%	87%
44	87%	87%	87%	87%	87%	87%
45	87%	86%	91%	41%	36%	91%
46	90%	87%	90%	89%	66%	89%
47	73%	73%	81%	85%	73%	85%
48	84%	84%	84%	84%	84%	87%
49	84%	92%	73%	92%	38%	92%
50	85%	89%	86%	85%	85%	87%
51	73%	72%	77%	83%	76%	91%
52	66%	74%	73%	79%	79%	82%
53	76%	49%	48%	76%	76%	76%
54	72%	83%	82%	85%	85%	87%
55	79%	87%	86%	79%	78%	100%
56	77%	80%	74%	73%	86%	87%
57	69%	80%	80%	86%	86%	90%
58	100%	67%	100%	67%	100%	100%
59	74%	79%	85%	87%	79%	88%
60	80%	80%	71%	100%	76%	100%

We compared the accuracy between single expert and our proposed fusion solution as shown in Figure 6.12, we can clearly see that the fusion solutions can obviously keep or improve the response accuracy based on different case studies, especially when some single experts proposed wrong solutions. It is because our proposed knowledge fusion is designed based on R-V consensus algorithm from a group of target consensus nodes, which can make an agreement in a secure and efficient way.

In addition, we also test the execution time to process multiple responses for each request. Figure

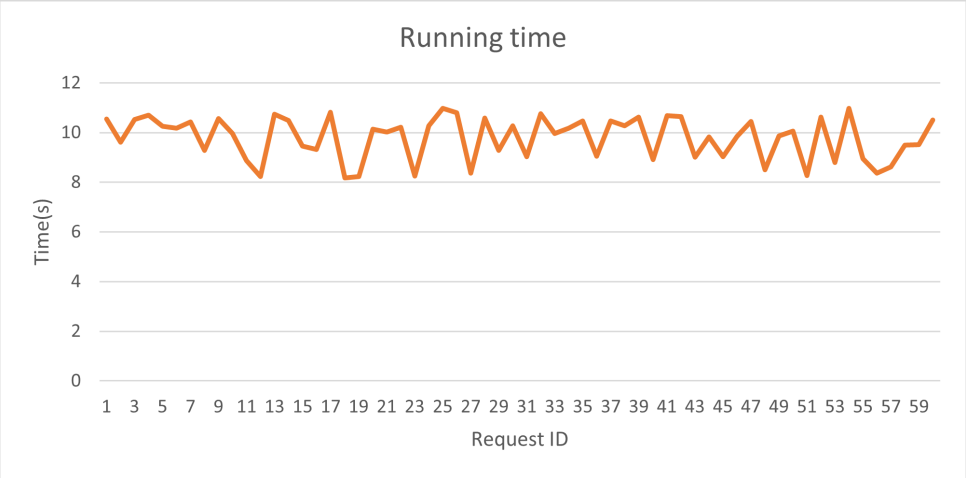


Figure 6.13: The execution time of our proposed fusion scheme in conversation system.

6.13 presents the running time from sending the request to getting the final fusion solution. From the test results, we can also see that the overall running time of all requests are within 11s when we set the maximum prescribed time as 10s, which means the processing time of fusion scheme based on our proposed consensus algorithm is less than 1s, and it also makes consistent with the experimental results in Chapter 5.

To sum up, it can be seen that the proposed decentralized fusion scheme has a greater effect including both accuracy and speed, which can prove the feasibility of smart-contract enabled decentralized knowledge fusion for blockchain-based conversation system.

6.6 Summary

In this chapter, we presented a novel blockchain-based framework for conversation system based on a decentralized knowledge fusion scheme by blockchain smart contracts, then designed the concrete decentralized knowledge fusion scheme to depict the protocol logic. A series of algorithms based on blockchain smart contracts were proposed to construct our proposed scheme under the novel framework. Meanwhile, we analysed how our decentralized knowledge fusion can handle centralized problems and make security. Finally, three case studies with 60 test cases used to validate the feasibility and effectiveness of the proposed approach. There are two main advantages of the designed system: 1) The decentralized knowledge fusion is highly secured and reliable as all the contents are recorded and stored on blockchain. 2) The conversation system with the decentralized knowledge fusion makes the interaction accuracy improving within a reasonable time.

Chapter 7. Conclusion and Future Directions

7.1 Summary

The purpose of this study is to design and integrate a blockchain-based decentralized conversation system among multiple experts to answer the core research question:

RQ: How can blockchain technology be integrated into conversation systems to provide security and trustworthy conversational services?

The research results are framed in the context of answering the research questions proposed in Section 1.5. Table 7.1 summarize the research results for each sub-research question.

Table 7.1: Summary of the research results of the study

SRQ1.1 *What is research landscape of blockchain technology in the conversation system?*
Research results: The current conversation system is physically decentralized; however, it contains critical components such as text processing, knowledge management and data storage that use large, centralized conversational services. This has some limitations and challenges that need to be addressed, especially considering security issues, such as data obtained from non-verified sources. Many sources rely on their own data and there is lack of security for private data. By analysing the benefits of blockchain compared with traditional centralized based mechanism, we have presented the vision of building a blockchain-based conversation system which has been designed to achieve security and trustworthy conversational services.
SRQ1.2 *What are key requirements of building a decentralized conversation system based on the blockchain technology to reach its fulfill potential?*
Research results: Layered security conversation system architecture can be built using the blockchain technology. Therefore the key requirements are identified from many aspects including database security, protocol design, application, and contracts requirement.
SRQ2.1 *What indexes (such as centralization degree, resource usage, etc.) should be considered to evaluate different platforms?*

Table 7.1 continued from previous page

Research results: Based on the requirements analysis of the conversation system, which includes content reliability and confidentiality, immediacy response, open-ended and extensible, four group decision indexes, including decentralized architecture, storage and sharing, computing performance, and scalability have been selected to match our design aim, and the detailed items are identified as well. Then for decision items of each group, the corresponding blockchain configurations have been chosen as multiple criteria for evaluating blockchain platforms.
SRQ2.2 *How can the applicability evaluation model be built?*
Research results: The applicability evaluation model is built based on the hierarchy structure. The multiple criteria are selected from group decision indexes, whereas the alternative platform options to be evaluated are Ethereum, Fabric, Corda, Multichain, etc. Applicability levels are defined as very inappropriate, inappropriate, appropriate, and very appropriate. 1) Cosidering each group of decision index, the blockchain platform selection with multiple criteria are formulated. 2) Weighted membership matrixes of each blockchain platform are built by using multiple criteria. 3) The three most popular weighting methods including AHP, Fuzzy based AHP and TOPSIS are utilized to combine and evaluate the criteria weighting. Thereafter, the final decision result will be judged by consistent verification and obtain the best fitting blockchain platform. Based on the evaluation results comparing the final weights of each criterion using AHP, FAHP, and FTOPSIS, it shows that the top three criteria are consistent with these three methods. Hence, the consensus protocol, chain structure, storage, and sharing are the most important considerations when selecting blockchain platforms for the conversation system. Furthermore, the evaluation results also show that there are no differences between the rankings of the alternatives using these three methods, and Hyperledger Fabric is the first choice using for conversation system compared to Ethereum, Corda, and Multichain.
SRQ3.1 *What should be the blockchain structure and block content?*
Research results: A master-slave blockchain is proposed to support transaction concurrency among different domains in the conversation system. The master-slave chain contains two layers. One layer comprises slave chains with different conversation domains, and another layer is a master chain, which is used to realize cross-chain authentication and to maintain the global consistency of transactions. The master chain is constructed using verification blocks, whereas the slave chain comprises domain blocks. For the block contents, The domain block contains details of the intra-chain transactions, and the verification blocks in the master chain will store a summary of the domain blocks within a time frame, and ensure inter-chain certifications during this time frame.
SRQ3.2 *How are the linking and validation mechanisms between chains designed?*

Table 7.1 continued from previous page

Research results: The linking and validation mechanisms based on hash anchoring are presented to link master and slave chains. In the body of the verification block, the summary of the domain blocks is stored. Therefore, any validations of the transactions in the domain blocks will also alter the master chain, which guarantees the tamper-proof nature of the blockchain.
SRQ3.3 *What metrics will be used to assess the efficiency of the proposed blockchain structure?*
Research results: The evaluation metrics of the proposed master-slave chain, including proof of Feasibility and transaction throughout are utilized to verify the applicability of the proposed master–slave chain structure. 1) Based on the presented two theorems and proofs, it is clear that the proposed master-slave chainis feasible to keep data consistent and prevent possible attacks. 2) To test the throughout, we deploy Hyperledger Blockchain 2.0 into Cloud, which has one master chain and three slave chains with one orderer service, three organizations and a total of 100 nodes. We compared the experimental results using same consensus algorithm in both single chain and the proposed master-slave using the same number of nodes. Based on the experimental results, there are no obvious difference between single domain and multiple domains of the single blockchain model, whereas considering the master-slave chain model, it can be concluded that the transaction throughput is related to the number of slave chains with different domains. This indicates that the more slave chains are, the greater the transaction throughput performance. To sum up, it has been confirmed that the proposed master-slave chain model can concurrently generate blocks and process different types of domain transactions, which is an effective way to improve the throughput of blockchain transactions.
SRQ4.1 *How is consensus and incentive mechanism for sharing and maintaining conversation scenarios designed?*

Table 7.1 continued from previous page

Research results: A hybrid consensus protocol based on reputation-driven voting is presented to ensure security and consistent collaboration among slave and master chain. 1) Consensus based on reputation-driven voting is designed and applied in each slave chain (R-V consensus), which includes the selection of target consensus nodes, consensus processing and consensus confirmation. 2) A dynamic construction strategy for master nodes is presented in the master chain. To select the master nodes, the evaluation value will be considered based on its reputation, computing power, transaction activity, and times selected as the master node. 3) The global PBFT consensus has been used in master chain, which removes the first phase "Pre-prepare" from the client. This is because our master chain is used to verify the inter-chain access authentication and is used as backup of the domain blocks. This does not involve the sorting of requests and only focuses on verification.
SRQ4.2 *Which incentive scheme is suitable for a conversation system?*
Research results: The incentive scheme including both the economic-based model and non-economic model was presented based on the proposed consensus protocol. This compensates participants with both financial and non-financial rewards in the conversation system. 1) The economic-based model is based on an issuance mechanism, determined by the total number of tokens, allocation ratio (domain and master nodes), and attenuation parameters. 2) The non-economic reward of each node is calculated according to the transaction activity (TA) and reputation value (RV) based on multiple linear regressions.
SRQ4.3 *What metrics will be used to assess the efficiency of the proposed protocol?*
Research results: Three evaluation metrics are used to assess the efficiency of the proposed consensus protocol, including transactions per second (TPS), delays, and fault tolerance. To perform our evaluation continuously and quickly, The transaction simulation module consisting of two types transactions in the experiments are used: one type is an inter-chain transaction with conversational rules and contribution assignment, another type is the intra-chain transaction with cross-chain authentication. Each transaction is created using a registered blockchain account, signed with its private key.

Table 7.1 continued from previous page

To verify the efficiency of the proposed consensus technique for the master–slave chains, we assumed the experimental results when using the classical PBFT as the ground truth for our master-slave chain. We have built the two test networks to verify the performance of the proposed consensus protocol.

1) Considering the same total number of nodes in test network one, the average number of verified transactions is approximately 167 per second in the classical PBFT and 182 per second in our proposed consensus. Nevertheless considering the test network two, the average number of verified transactions is approximately 151 per second in the classical PBFT and 168 per second in our proposed consensus.

2) The experimental results for the average delay of ten blocks shown that the block generation time of our proposed algorithm is approximately 0.65s in test network one whereas the classical PBFT is about 0.76s, and in test network two, the delay performance of our proposed algorithm is 1.17s and classical PBFT is approximately 1.29s.

3) The experiments for fault tolerance show that if the number of fault nodes in the slave and master are greater than the maximum number, the malicious transaction and block cannot be approved by our proposed consensus protocol.

4) To validate the overall feasibility and effectiveness of the proposed blockchain-based conversation system, these case studies with 60 samples are utilized to test the smart contracts-enabled knowledge fusion in the conversation system based on the proposed master-chain structure and consensus protocol. The experimental results show that the fusion solutions can obviously improve the response accuracy based on the different case studies, especially when some experts proposed wrong solutions. This is because our proposed knowledge fusion is designed based on the R-V consensus algorithm from a group of target consensus nodes, which can make an agreement in a secure and efficient way. We also tested the overall running time, and the results show all the requests within 11s when we set the maximum prescribed time as 10s. This indicated that the processing time of the decentralized fusion scheme based on our proposed consensus algorithm is less than 1s. Thus, the proposed decentralized conversation system has a greater effect for both accuracy and speed.

To sum up, our proposed consensus can improve the TPS performance and reduce the average delay compared to the classical PBFT in the Fabric network. This is because we define the minimum requirement to set the target consensus nodes, which is an efficient way to reduce the transmission and verification time. Moreover, we remove the pre-prepare phase initiated by the client and set the time interval to initiate a consensus, which simplifies three-phase PBFT to two phases. Furthermore, the proposed decentralized knowledge fusion is highly secured and reliable because all the contents are recorded and stored on the blockchain. The conversation system with the decentralized knowledge fusion can improve the interaction accuracy within a reasonable time.

Having reached the research results of each sub research questions, now the answer can be given for the core research question:

RQ: *How to integrate blockchain technology into conversation system to provide security and trustworthy conversational services?*

Answer: This study has shown that by designing and implementing master-slave chain structure with a hybrid consensus protocol, an incentive scheme considering both financial and non-financial rewards and smart contract-enabled knowledge fusion, the blockchain-based decentralized conversation system can provide various conversational services in a security and trustworthy manner.

7.2 Significant findings

This thesis began with the introduction of a conversation system which aims to provide various efficient and trustworthy conversational services. Many challenges are pointed out related to the current centralized conversation management and the decentralized conversation management are urgently needed. After integrating the blockchain technology into conversation system in this study, there are some significant findings, which are as follows:

1) **The vision of building blockchain-based conversation system architecture**: Based on the literature review in Chapter 2, we have conducted a comprehensive survey on the current conversation system architectures and blockchain technologies to understand the challenges of the conversation system architectures and the benefits of blockchains compared to traditional security mechanism. This study presents the vision of building a blockchain-based conversation system architecture which has been designed to achieve efficient and trustworthy conversational services in a secure manner. The study discusses the key technical requirements from different aspects related to the proposed architecture, and analyzes the trends and challenges mapping to these key requirements. This finding guides more detailed and innovative solutions to implement blockchain-based decentralized conversation system.

2) **Selection of the best fitting blockchain platform for the conversation system**: Based on the requirement analysis of the conversation system, the selection of blockchain platforms has been modelled as a decision-making problem with formulated multiple criteria regarding the identified requirement analysis and design aims. Hyperledger Fabric was selected as the best fitting blockchain

platform for the conversation system using multiple measurements and consistent evaluation analysis. This finding is the foundation for designing, implementing and evaluating blockchain-driven solutions.

3) **A master–slave chain model for conversation system:** Based on the analysis of blockchain interoperability, the study proposes a master-slave chain to process multiple conversation interactions concurrently from different domains.Considering our design, each slave chain represents a single conversation domain, whereas the master chain is utilized as backup of the summary of slave chains and to approve the inter-chain authentication. Therefore, it can ensure data consistency with the slave chains. The proposed structure was evaluated using non-tamper proofs and transactions throughout, confirming that the master-slave chain is an effective way to concurrently generate blocks and process different types of domain transactions.

4) **A hybrid consensus algorithm based on the master-slave chain**: We examined on the bottleneck of existing consensus protocols in the Hyperledger Fabric and presented an algorithm optimisation suitable for our proposed master-chain in a conversation system. The consensus, based on reputation-driven voting and global PBFT, is utilized for intra and inter-chain verifications respectively. Furthermore, an incentive scheme is designed to compensate the participants using both financial and non-financial rewards for all the participants based on our proposed consensus protocol. The economic-based model is based on an issuance mechanism determined by the total number of tokens, allocation ratio (domain and master nodes), and attenuation parameters, whereas the non-economic model focuses on how to provide the participants with more opportunities to obtain good content considering the reputation values and transaction activities. We evaluated the proposed consensus protocol using TPS, delay and fault tolerance to prove the efficiency of the proposed consensus algorithm for master–slave chains.

5) **Smart contract enabled decentralized knowledge fusion**: We investigated the problems of the traditional knowledge fusion construction and presented a novel blockchain-based framework for the conversation system based on decentralized knowledge fusion. A concrete scheme using blockchain smart contracts are utilized to secure the knowledge fusion process and to assign the contributions of multiple experts. The proposed knowledge fusion scheme is designed and implemented based on the proposed master-slave chain and consensus protocol. The feasibility and effectiveness of the proposed approach are verified using the accuracy and execution time.

In summary, the integration of blockchain technology into conversation system has considered both

security and trustworthy to support the various conversational services. This study has demonstrated the feasibility and effectiveness of the proposed blockchain-based decentralized conversation system for multiple case studies.

7.3 Future directions

In this study, the problems of unfair incentive schemes, contributions and conversation content tampering and privacy aspect in conversation systems are addressed. However, a number of directions remain open issues that could be extended for future research.

7.3.1 Decision-making model with different blockchain platforms

In the study, we consider how quickly organize a new market blockchain platform into our proposed decision-making model. However, the new blockchain platforms may include some new features that may affect requirement analysis, design aims and multiple criteria selection. In addition, new measurement methods may also be needed to add to the presented decision model for further study.

7.3.2 Various big-data verification and storage

As a fundamental and key component of blockchains, research on the consensus mechanism is crucial. In addition to the security and efficiency issues discussed in our research, many problems still require research attention. For instance, a potential high capacity (Singh et al. 2020) exists in the master chain if there are a large number of slave chains. Furthermore, our experiments are focused on text-based data, but real conversation systems may also involve image or video data, thus, techniques to handle various types of big-data verification and manage consistent storage are technical problems that should be considered in the future.

7.3.3 Scalability of blockchain-based conversation system

Blockchain technology is still in its early stage and there are several meaningful works which can be explored in the future. For example, our current design was focused on approving feasibility and improving security. However, conversation systems should also consider scalability, especially along with the more nodes involving. Our experiments tested a maximum of 100 nodes with thousands of

transactions; this is still relatively small compared with large scale conversational services. Thus, we may need to consider more efficient algorithms in the future. In addition, more evaluation applications may also need to be tested in order to make our research more flexible and practical.

7.3.4 Security enhancement of blockchain itself

Although blockchain is based on secure technology, a blockchain needs to be protected as well, and currently some security risks behind blockchain stay cautions as well. When using blockchain, users need to register and generate their private keys to encrypt transactions with digital signatures. Some researchers have investigated vulnerabilities in RSA and ECDSA (Mayer 2016, Mahto et al. 2016), which are typical digital signature methods used in blockchains. They found criminals could recover the user's private key, and then the user's blockchain account will face the high risk of being tampered by others. In addition, as programmable and executable code in blockchain, smart contracts may also have security issues caused by program defects. Many types of security issues of smart contracts have been discovered, such as exception disorder,immutable and randomness bug,stack overflow, unpredictable state, etc (Lin & Liao 2017, Peng et al. 2021). Therefore, it is essentially to have further studies on the security enhancement solutions of blockchain technology.

7.3.5 More machine learning adoption in blockchain smart contracts

In our study, we integrated several nature language processing techniques such as noun phrases extraction, POS tagging and text similarity, etc., to support a decentralized knowledge fusion scheme. To handle more complex conversation sensations in the future, an intelligent agent raises the importance of machine learning technologies. For example, we can construct our conversation system by incorporating recommendations to provide adaptive and personalized interactions. Another example is the integration of image retrieval into a conversation system to perform high-level semantic concepts according to the user's intent. The more the combination of machine learning and blockchain, the more intelligent the conversation system. Thus, there is a need to study machine learning adoption in blockchain smart contracts to make blockchain-based conversation systems more intelligent and resilient.

Appendix A. Chapter 1 Appendix

Recently, human computer conversation has attracted increasing attention due to its promising potentials and alluring commercial values. With the development of big data and AI techniques, the goal of creating an automatic human computer conversation system, as our personal assistant or chat companion, is no longer an illusion(CHEN et al. 2017).

The conversation system can be classified by two types: Chat-oriented (open-domain) system and task-oriented (closed-domain) system. Task-oriented systems are created to solve a particular problem: find the information requested by a user, accomplish a task. Open-domain systems are not limited to one domain, they are meant to be omni-purpose: e.g. Siri is supposed to do anything that can be done by an iPhone.

A.1 Open Domain Conversation Systems

A Dialogue-Based Computer-Assisted Second Language Learning (DB-CALL) system is built in a chatbot form which is engaged into conversations with uses in given scenarios (Huang et al. 2017). Since in scenarios conversations, meaningful expression could be comprehended as errors if is not included in scenarios. Huang et al. (2017) state that utilising a conversation corpus search engine, the problem of conversations out of scenarios could be simultaneously addressed, which are the problems of conversations out of scenarios and stimulate user learning interest. While the concept of using open domain conversation system has its advantages, the GenieTutor Plus Huang et al. (2017) utilised has an 33.33% 'Turn success ratio (non-topic)' (Huang et al. 2017). As for the context of topic conversation, users are inclined to assess non-topic responses in a stricter manner. This brings a contrary user experience when encountered with an open domain conversation system that is designed to improve useability satisfaction to increase user's interest in learning.

Even though presently spoken conversation systems function comparatively well in closed domains

in which interaction topics are acknowledged beforehand and in which the wording users are expected to use could be predetermined. SDS are not so prosperous for interaction in open domains, in which user might talk freely about anything as they please (Hirschberg & Manning 2015). There are also conversation systems which use machine learning techniques to extend the quantities of database queries to cover maximum open domain conversation interactions with users. Sordoni et al. (Sordoni et al. 2015) introduced a system which generates novel responses that are trained on many Twitter conversations. The conversations are unstructured and utilise a neural network architecture to tackle issues of sparsity which emerge when contextual data is integrated into statistic models. Their research found that the system considers prior conversational utterances and the generative models with dynamic-context demonstrate constant accumulation with context and non-context sensitive Information Retrieval bases and machine translation(Sordoni et al. 2015). From the outcomes perspective, machine learning method conversation systems are expected to be developed with larger and more comprehensive datasets with trillions of dialogue-sets instead of focusing on improving satisfaction of user experience.

The classification of building a conversation system to be open or close domain would not be a top priority in developing an intelligent conversation system as a language tutor. Since users' satisfaction and their willingness to spend more time interacting with the system is a key element for improving their English skills and fulfilling system's role as a language tutor.

A.2 Close Domain Conversation Systems

Some conversation systems aiming to assist students to learn English have been developed, one of which is named GenieTutor by Korea Electronics and Telecommunications Research Institute. GenieTutor is classified as a close domain conversation system since it generates questions to students based on a specific topic, communicates with the user according to the particular topic scenario, and generates feedback if there are grammatical errors from the user (Choi et al. 2017).

Some apparent disadvantages of GenieTutor that limits conversation fluency of the learner and cannot perform conversations freely if the utterance is not included in the topic were resolved by Choi et al. They introduced an upgrade to the system called GenieTutorPlus, which could have free conversation with users outside of topics and provide feedback on any detected grammar mistakes of the learner jeopardising conversation fluency. This system is designed to respond to out-of-topic utterance and topic sentences with response from chatbot and topic conversation database respectively and is evaluated by 'the average success rate of the conversation turn' and other rates (Choi et al. 2017).

Table A.1: The comparisons between different types of conversation systems

Area	Chat	Knowledge	Task	Recommendation
Objective	Chatting	Knowledge acquisition	Complete specific task	Information recommender
Type	Open domain	Open domain	Closed domain	Closed domain
Turn number	The more the better	The less the better	The less the better	The less the better
Application	Entertainment Emotional communication, etc.	Custom-service, education, etc.	Virtual personal assistant, etc.	Personal recommendation.
Typical samples	Siri	Watson, Wolfram, Alpha	Cortana, Allo, etc.	Quartz

There are many closed domain conversation systems are used for intelligent tutoring and teaching. Intelligent Tutoring and teaching systems are software agent AI systems, the task of which is to interactively tutor students as an imitation human teacher (Franklin 2014). A conversational intelligent tutoring system (CITS) could predict and dynamically adjust to a learner's studying style (Latham 2012). A further development of intelligent tutoring systems is question-based conversation system which are designed to provide a question scenario and facilitate students to advance their learning evolution through the question, such as (Kwon et al. 2015), (Franklin 2014), etc.

Based on the above discussion, we can according to the main applied areas to summarize key features of existing open-domain and closed-domain conversation systems as Table A.1.

Appendix B. Chapter 2 Appendix

The existing blockchain-related academic papers are mainly reviewed from four primary areas: constructive technologies for blockchain, applications for blockchain, evaluation and opportunities as shown in B.1.

Table B.1: Summarization of current research topics related to blockchain technology

Research problem	Objective	Key points	References
Constructive technologies for blockchain	improving the current components of blockchain	data structure design security enhancing and privacy protection consensus protocol improvement	(Gramoli 2017, Hughes n.d.) (Zyskind et al. 2015, Kosba et al. 2016) (Duffield & Hagan 2014, Milutinovic et al. 2016, Pass & Shi 2017)
Applications for blockchain	improving previous application, creating new application and designing smart contracts for different applications	Finance IoT Public and social services Cloud Services Other Internet services	(Guo & Liang 2016, Nguyen 2016) (Dorri, Kanhere, Jurdak & Gauravaram 2017, Dorri, Kanhere & Jurdak 2017, Song et al. 2018) (Larimer et al. 2016, Chakravorty & Rong 2017) (Liang et al. 2017, Gaetani et al. 2017, Zhang et al. 2018) (Lu et al. 2018, Li et al. 2018, Viriyasitavat et al. 2018, Su et al. 2018)
Evaluation and challenges	evaluating of blockchain platforms and analyzing future trends and challenges	evaluating of blockchain platforms trends and challenges	(Idelberger et al. 2016, Aniello et al. 2017, Dinh et al. 2018) (Münsing et al. 2017, Fridgen et al. 2018, Luu et al. 2016, Atzei et al. 2017)

Constructive technologies for blockchain: this section focuses on improving the current compo-

nents of blockchain such as data structure design, security enhancement and privacy protection as well as current consensus protocol improvement. The research on data structure was firstly based on hash-tables, however with the significant growth of blockchain usage, several new data structures with scalable, light-weight and decentralized features were proposed. In this regard, Directed Acyclic Graph (DAG) for maintaining transaction information and RadixDLT for scaling linearly in an unbounded and efficient manner are the proposed structures. Some researchers have discussed how to make a possible solution using blockchain for building mutual trust within society. For example, an automated manager without any third-party intervention was presented to turn a blockchain into access control. The decentralized system was proposed to retain transactional privacy from public view using cryptographic primitives such as zero-knowledge proofs. In addition, many researchers focus on consensus protocols, such as the improvement of the performance and efficiency of existing protocols as well as the creation of new consensus protocols.

Applications for blockchain: there are many papers which discuss improving previous applications, creating new applications, while designing smart contracts for different applications represents another key hot topic. Since a huge amount of the current Internet services are developed in a centralized manner, researchers have tried to explore decentralized structures to deal with increasing security problems and limitations of the current Internet services. Except for the initial financial applications, more research focusing on some certain areas related to Internet services, such as the Internet of Things(IoT) (Conoscenti et al. 2016), public and social services (Chakravorty & Rong 2017), cloud services (Xia et al. 2017) and other Internet services such as reputation (Dennis & Owen 2015) and crowdsourcing (Li et al. 2018) are also being conducted.

Evaluation and challenges: since blockchain combines multiple technologies to ensure an immutable, irrevocable and traceable ledger, there are some related works centred on evaluating and analyzing the overhead and performance of the proposed decentralized architecture, including throughput and latency, scalability, fault tolerance, protocol and network security. On the basis of evaluation, some challenges about current blockchain platforms can be found, such as storage capacity of blockchain, the process of automation, the security and efficiency of smart contracts and so on.

Appendix C. Chapter 4 Appendix

Fabric Network Configuration

1. Network Structure

One orderer, three organizations, and each organization has three peers.

Org Name	Org ID
Org1	Org1MSP
Org2	Org2MSP
Org3	Org3MSP

1.1Generate Certificate

(1)Generate certificates

```
sudo vim crypto-config.yaml
```

```
OrdererOrgs:
  - Name: Orderer
    Domain: example.com
    Specs:
      - Hostname: orderer
PeerOrgs:

  - Name: Org1 #org name
    Domain: org1.example.com #org domain
    EnableNodeOUs: true
    Template:
      Count: 3  #the number of peer
    Users:
      Count: 1 #the number of user

  - Name: Org2
    Domain: org2.example.com
    EnableNodeOUs: true
    Template:
      Count: 3
    Users:
      Count: 1

  - Name: Org3
    Domain: org3.example.com
    EnableNodeOUs: true
    Template:
      Count: 3
    Users:
      Count: 2
```

(2)Generate configuration file of certificates

```
bin/cryptogen generate --config=crypto-config.yaml
```

```
root@admin:~/hyfa/network# bin/cryptogen generate --config=crypto-config.yaml
org1.example.com
org2.example.com
org3.example.com
root@admin:~/hyfa/network#
```

2.Build a genesis block and channel configuration

(1)Generate genesis block file

```
sudo vim configtx.yaml
```

```
# Copyright IBM Corp. All Rights Reserved.
#
# SPDX-License-Identifier: Apache-2.0
#

---
################################################################################
#
#   Section: Organizations
#
#   - This section defines the different organizational identities which will
#   be referenced later in the configuration.
#
################################################################################
Organizations:

    - &OrdererOrg
        Name: OrdererOrg
        ID: OrdererMSP
        MSPDir: crypto-config/ordererOrganizations/example.com/msp
        Policies:
            Readers:
                Type: Signature
                Rule: "OR('OrdererMSP.member')"
            Writers:
                Type: Signature
                Rule: "OR('OrdererMSP.member')"
            Admins:
                Type: Signature
                Rule: "OR('OrdererMSP.admin')"
        OrdererEndpoints:
            - orderer.example.com:7050

    - &Org1
        Name: Org1MSP
        ID: Org1MSP
        MSPDir: crypto-config/peerOrganizations/org1.example.com/msp

        Policies:
            Readers:
```

```
                Type: Signature
                Rule: "OR('Org1MSP.admin', 'Org1MSP.peer', 'Org1MSP.client')"
            Writers:
                Type: Signature
                Rule: "OR('Org1MSP.admin', 'Org1MSP.client')"
            Admins:
                Type: Signature
                Rule: "OR('Org1MSP.admin')"
            Endorsement:
                Type: Signature
                Rule: "OR('Org1MSP.peer')"

        AnchorPeers: #Achor peer
            - Host: peer0.org1.example.com
              Port: 7051

    - &Org2
        Name: Org2MSP
        ID: Org2MSP
        MSPDir: crypto-config/peerOrganizations/org2.example.com/msp

        Policies:
            Readers:
                Type: Signature
                Rule: "OR('Org2MSP.admin', 'Org2MSP.peer', 'Org2MSP.client')"
            Writers:
                Type: Signature
                Rule: "OR('Org2MSP.admin', 'Org3MSP.client')"
            Admins:
                Type: Signature
                Rule: "OR('Org2MSP.admin')"
            Endorsement:
                Type: Signature
                Rule: "OR('Org2MSP.peer')"

        AnchorPeers:
            - Host: peer0.org2.example.com
              Port: 10051

    - &Org3
        Name: Org3MSP
        ID: Org3MSP
        MSPDir: crypto-config/peerOrganizations/org3.example.com/msp

        Policies:
            Readers:
                Type: Signature
                Rule: "OR('Org3MSP.admin', 'Org3MSP.peer', 'Org3MSP.client')"
            Writers:
                Type: Signature
                Rule: "OR('Org3MSP.admin', 'Org3MSP.client')"
            Admins:
                Type: Signature
                Rule: "OR('Org3MSP.admin')"
            Endorsement:
                Type: Signature
                Rule: "OR('Org3MSP.peer')"

        AnchorPeers:
```

```
            - Host: peer0.orgslave3.example.com
              Port: 13051

################################################################################
#
#   SECTION: Capabilities
#
#   - This section defines the capabilities of fabric network. This is a new
#   concept as of v1.1.0 and should not be utilized in mixed networks with
#   v1.0.x peers and orderers.  Capabilities define features which must be
#   present in a fabric binary for that binary to safely participate in the
#   fabric network.  For instance, if a new MSP type is added, newer binaries
#   might recognize and validate the signatures from this type, while older
#   binaries without this support would be unable to validate those
#   transactions.  This could lead to different versions of the fabric binaries
#   having different world states.  Instead, defining a capability for a channel
#   informs those binaries without this capability that they must cease
#   processing transactions until they have been upgraded.  For v1.0.x if any
#   capabilities are defined (including a map with all capabilities turned off)
#   then the v1.0.x peer will deliberately crash.
#
################################################################################
Capabilities:
    # Channel capabilities apply to both the orderers and the peers and must be
    # supported by both.
    # Set the value of the capability to true to require it.
    Channel: &ChannelCapabilities
        # V2_0 capability ensures that orderers and peers behave according
        # to v2.0 channel capabilities. Orderers and peers from
        # prior releases would behave in an incompatible way, and are therefore
        # not able to participate in channels at v2.0 capability.
        # Prior to enabling V2.0 channel capabilities, ensure that all
        # orderers and peers on a channel are at v2.0.0 or later.
        V2_0: true

    # Orderer capabilities apply only to the orderers, and may be safely
    # used with prior release peers.
    # Set the value of the capability to true to require it.
    Orderer: &OrdererCapabilities
        # V2_0 orderer capability ensures that orderers behave according
        # to v2.0 orderer capabilities. Orderers from
        # prior releases would behave in an incompatible way, and are therefore
        # not able to participate in channels at v2.0 orderer capability.
        # Prior to enabling V2.0 orderer capabilities, ensure that all
        # orderers on channel are at v2.0.0 or later.
        V2_0: true

    # Application capabilities apply only to the peer network, and may be safely
    # used with prior release orderers.
    # Set the value of the capability to true to require it.
    Application: &ApplicationCapabilities
        # V2_0 application capability ensures that peers behave according
        # to v2.0 application capabilities. Peers from
        # prior releases would behave in an incompatible way, and are therefore
        # not able to participate in channels at v2.0 application capability.
        # Prior to enabling V2.0 application capabilities, ensure that all
        # peers on channel are at v2.0.0 or later.
        V2_0: true
```

```
################################################################################
#
#   SECTION: Application
#
#   - This section defines the values to encode into a config transaction or
#   genesis block for application related parameters
#
################################################################################
Application: &ApplicationDefaults

    # Organizations is the list of orgs which are defined as participants on
    # the application side of the network
    Organizations:

    # Policies defines the set of policies at this level of the config tree
    # For Application policies, their canonical path is
    #   /Channel/Application/<PolicyName>
    Policies:
        Readers:
            Type: ImplicitMeta
            Rule: "ANY Readers"
        Writers:
            Type: ImplicitMeta
            Rule: "ANY Writers"
        Admins:
            Type: ImplicitMeta
            Rule: "MAJORITY Admins"
        LifecycleEndorsement:
            Type: ImplicitMeta
            Rule: "MAJORITY Endorsement"
        Endorsement:
            Type: ImplicitMeta
            Rule: "MAJORITY Endorsement"

    Capabilities:
        <<: *ApplicationCapabilities
################################################################################
#
#   SECTION: Orderer
#
#   - This section defines the values to encode into a config transaction or
#   genesis block for orderer related parameters
#
################################################################################
Orderer: &OrdererDefaults

    # Ordering node algorithm
    OrdererType: etcdraft

    # generate block per 2 seconds
    BatchTimeout: 2s

    BatchSize:

        # Maximum transaction number 100
        MaxMessageCount: 100
        # Maximum block size
        AbsoluteMaxBytes: 32 MB
        PreferredMaxBytes: 512 KB
```

```
    Organizations:

    Policies:
        Readers:
            Type: ImplicitMeta
            Rule: "ANY Readers"
        Writers:
            Type: ImplicitMeta
            Rule: "ANY Writers"
        Admins:
            Type: ImplicitMeta
            Rule: "MAJORITY Admins"
        BlockValidation:
            Type: ImplicitMeta
            Rule: "ANY Writers"

################################################################################
#
#   CHANNEL
#
#   This section defines the values to encode into a config transaction or
#   genesis block for channel related parameters.
#
################################################################################
Channel: &ChannelDefaults
    # Policies defines the set of policies at this level of the config tree
    # For Channel policies, their canonical path is
    #   /Channel/<PolicyName>
    Policies:
        # Who may invoke the 'Deliver' API
        Readers:
            Type: ImplicitMeta
            Rule: "ANY Readers"
        # Who may invoke the 'Broadcast' API
        Writers:
            Type: ImplicitMeta
            Rule: "ANY Writers"
        # By default, who may modify elements at this config level
        Admins:
            Type: ImplicitMeta
            Rule: "MAJORITY Admins"

    # Capabilities describes the channel level capabilities, see the
    # dedicated Capabilities section elsewhere in this file for a full
    # description
    Capabilities:
        <<: *ChannelCapabilities

################################################################################
#
#   Profile
#
#   - Different configuration profiles may be encoded here to be specified
#   as parameters to the configtxgen tool
#
################################################################################
Profiles:
```

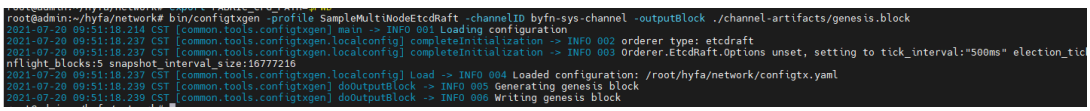
```
    TwoOrgsChannel:
        Consortium: SampleConsortium
        <<: *ChannelDefaults
        Application:
            <<: *ApplicationDefaults
            Organizations:
                - *Org1
                - *Org2
                - *Org3
            Capabilities:
                <<: *ApplicationCapabilities

    SampleMultiNodeEtcdRaft:
        <<: *ChannelDefaults
        Capabilities:
            <<: *ChannelCapabilities
        Orderer:
            <<: *OrdererDefaults
            OrdererType: etcdraft
            EtcdRaft:
                Consenters:
                - Host: orderer.example.com
                  Port: 7050
                  ClientTLSCert: ./crypto-
config/ordererOrganizations/example.com/orderers/orderer.example.com/tls/server.crt
                  ServerTLSCert: ./crypto-
config/ordererOrganizations/example.com/orderers/orderer.example.com/tls/server.crt
            Addresses:
                - orderer.example.com:7050

            Organizations:
            - *OrdererOrg
            Capabilities:
                <<: *OrdererCapabilities
        Application:
            <<: *ApplicationDefaults
            Organizations:
            - <<: *OrdererOrg
        Consortiums:
            SampleConsortium:
                Organizations:
                - *Org1
                - *Org2
                - *Org3
```

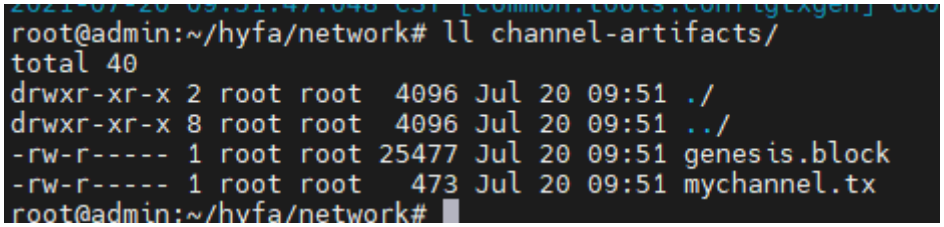
(2)Build a genesis block

```
mkdir channel-artifacts
export FABRIC_CFG_PATH=$PWD
bin/configtxgen -profile SampleMultiNodeEtcdRaft -channelID byfn-sys-channel -
outputBlock ./channel-artifacts/genesis.block
```



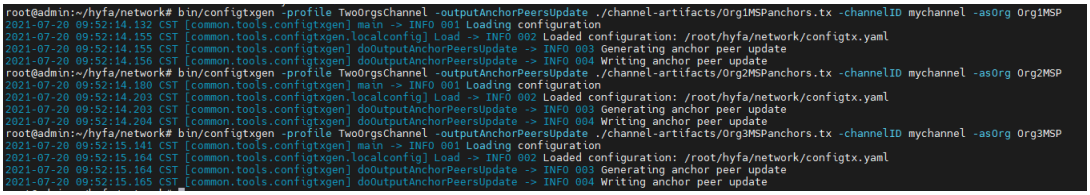
(3)Generate channel configuration

```
#generate channel
bin/configtxgen -profile TwoOrgsChannel -outputCreateChannelTx ./channel-
artifacts/mychannel.tx -channelID mychannel
#check generation status
ll channel-artifacts/
```



(4)Generate Anchor peer updating file

```
bin/configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-
artifacts/Org1MSPanchors.tx -channelID mychannel -asOrg Org1MSP
bin/configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-
artifacts/Org2MSPanchors.tx -channelID mychannel -asOrg Org2MSP
bin/configtxgen -profile TwoOrgsChannel -outputAnchorPeersUpdate ./channel-
artifacts/Org3MSPanchors.tx -channelID mychannel -asOrg Org3MSP
#check generation status
ll channel-artifacts/
```



3.docker-compose

(1)Configure base/docker-compose-base.yaml

```
gedit base/docker-compose-base.yaml
```

```
# Copyright IBM Corp. All Rights Reserved.
#
# SPDX-License-Identifier: Apache-2.0
#

version: '2'

services:


  orderer.example.com:
    container_name: orderer.example.com
    image: hyperledger/fabric-orderer
    environment:
```

```
      - ORDERER_GENERAL_LOGLEVEL=debug
      - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
      - ORDERER_GENERAL_GENESISMETHOD=file
      - ORDERER_GENERAL_GENESISFILE=/var/hyperledger/orderer/orderer.genesis.block
      - ORDERER_GENERAL_LOCALMSPID=OrdererMSP
      - ORDERER_GENERAL_LOCALMSPDIR=/var/hyperledger/orderer/msp
      # enabled TLS
      - ORDERER_GENERAL_TLS_ENABLED=true
      - ORDERER_GENERAL_TLS_PRIVATEKEY=/var/hyperledger/orderer/tls/server.key
      - ORDERER_GENERAL_TLS_CERTIFICATE=/var/hyperledger/orderer/tls/server.crt
      - ORDERER_GENERAL_TLS_ROOTCAS=[/var/hyperledger/orderer/tls/ca.crt]
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric
    command: orderer
    volumes:
        - ../channel-
artifacts/genesis.block:/var/hyperledger/orderer/orderer.genesis.block
        - ../crypto-
config/ordererOrganizations/example.com/orderers/orderer.example.com/msp:/var/hyperled
ger/orderer/msp
        - ../crypto-
config/ordererOrganizations/example.com/orderers/orderer.example.com/tls/:/var/hyperle
dger/orderer/tls
    ports:
      - '7050'


  peer0.org1.example.com:
    container_name: peer0.org1.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
    environment:
      - CORE_PEER_ID=peer0.org1.example.com
      - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:7051
      - CORE_PEER_CHAINCODEADDRESS=peer0.org1.example.com:7052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:7052
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer1.org1.example.com:8051
      #- CORE_PEER_GOSSIP_BOOTSTRAP=peer1.org1.example.com:9051
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer0.org1.example.com:7051
      - CORE_PEER_LOCALMSPID=Org1MSP
    volumes:
        - /var/run/:/host/var/run/
        - ../crypto-
config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer0.org1.example.com:/var/hyperledger/production
    ports:
      - '7051'
      - '7053'

  peer1.org1.example.com:
    container_name: peer1.org1.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
```

```
    environment:
      - CORE_PEER_ID=peer1.org1.example.com
      - CORE_PEER_ADDRESS=peer1.org1.example.com:8051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:8051
      - CORE_PEER_CHAINCODEADDRESS=peer1.org1.example.com:8052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:8052
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer1.org1.example.com:8051
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer0.org1.example.com:7051
      #- CORE_PEER_GOSSIP_BOOTSTRAP=peer0.org1.example.com:9051
      - CORE_PEER_LOCALMSPID=Org1MSP
    volumes:
        - /var/run/:/host/var/run/
        - ../crypto-
config/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer1.org1.example.com:/var/hyperledger/production

    ports:
      - '8051'
      - '8053'

  peer2.org1.example.com:
    container_name: peer1.org1.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
    environment:
      - CORE_PEER_ID=peer1.org1.example.com
      - CORE_PEER_ADDRESS=peer1.org1.example.com:9051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:9051
      - CORE_PEER_CHAINCODEADDRESS=peer1.org1.example.com:9052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:9052
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer1.org1.example.com:9051
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer0.org1.example.com:7051
      #- CORE_PEER_GOSSIP_BOOTSTRAP=peer0.org1.example.com:8051
      - CORE_PEER_LOCALMSPID=Org1MSP
    volumes:
        - /var/run/:/host/var/run/
        - ../crypto-
config/peerOrganizations/org1.example.com/peers/peer2.org1.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org1.example.com/peers/peer2.org1.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer2.org1.example.com:/var/hyperledger/production

    ports:
      - '9051'
      - '9053'

  peer0.org2.example.com:
    container_name: peer0.org2.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
    environment:
```

```
      - CORE_PEER_ID=peer0.org2.example.com
      - CORE_PEER_ADDRESS=peer0.org2.example.com:10051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:10051
      - CORE_PEER_CHAINCODEADDRESS=peer0.org2.example.com:10052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:10052
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer0.org2.example.com:10051
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer1.org2.example.com:11051
      - CORE_PEER_LOCALMSPID=Org2MSP
    volumes:
        - /var/run/:/host/var/run/
        - ../crypto-
config/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer0.org2.example.com:/var/hyperledger/production
    ports:
      - '10051'
      - '10053'

  peer1.org2.example.com:
    container_name: peer1.org2.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
    environment:
      - CORE_PEER_ID=peer1.org2.example.com
      - CORE_PEER_ADDRESS=peer1.org2.example.com:11051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:11051
      - CORE_PEER_CHAINCODEADDRESS=peer1.org2.example.com:10052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:11052
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer1.org2.example.com:11051
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer0.org2.example.com:10051
      - CORE_PEER_LOCALMSPID=Org2MSP
    volumes:
        - /var/run/:/host/var/run/
        - ../crypto-
config/peerOrganizations/org2.example.com/peers/peer1.org2.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org2.example.com/peers/peer1.org2.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer1.org2.example.com:/var/hyperledger/production
    ports:
      - '11051'
      - '11053'

  peer2.org2.example.com:
    container_name: peer2.org2.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
    environment:
      - CORE_PEER_ID=peer2.org2.example.com
      - CORE_PEER_ADDRESS=peer2.org2.example.com:12051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:12051
      - CORE_PEER_CHAINCODEADDRESS=peer2.org2.example.com:12052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:12052
```

```
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer2.org2.example.com:12051
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer0.org2.example.com:10051
      - CORE_PEER_LOCALMSPID=Org2MSP
    volumes:
        - /var/run/:/host/var/run/
        - ../crypto-
config/peerOrganizations/org2.example.com/peers/peer2.org2.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org2.example.com/peers/peer2.org2.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer2.org2.example.com:/var/hyperledger/production
    ports:
      - '12051'
      - '12053'

  peer0.org3.example.com:
    container_name: peer0.org3.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
    environment:
      - CORE_PEER_ID=peer0.org3.example.com
      - CORE_PEER_ADDRESS=peer0.org3.example.com:13051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:13051
      - CORE_PEER_CHAINCODEADDRESS=peer0.org3.example.com:13052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:13052
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer0.org3.example.com:13051
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer1.org3.example.com:14051
      - CORE_PEER_LOCALMSPID=Org3MSP
    volumes:
        - /var/run/:/host/var/run/
        - ../crypto-
config/peerOrganizations/org3.example.com/peers/peer0.org3.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org3.example.com/peers/peer0.org3.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer0.org3.example.com:/var/hyperledger/production
    ports:
      - '13051'
      - '13053'

  peer1.org3.example.com:
    container_name: peer1.org3.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
    environment:
      - CORE_PEER_ID=peer1.org3.example.com
      - CORE_PEER_ADDRESS=peer1.org3.example.com:14051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:14051
      - CORE_PEER_CHAINCODEADDRESS=peer1.org3.example.com:14052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:14052
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer1.org3.example.com:14051
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer0.org3.example.com:13051
      - CORE_PEER_LOCALMSPID=Org3MSP
    volumes:
        - /var/run/:/host/var/run/
```

```
        - ../crypto-
config/peerOrganizations/org3.example.com/peers/peer1.org3.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org3.example.com/peers/peer1.org3.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer1.org3.example.com:/var/hyperledger/production
    ports:
      - '14051'
      - '14053'

  peer2.org3.example.com:
    container_name: peer2.org3.example.com
    extends:
      file: peer-base.yaml
      service: peer-base
    environment:
      - CORE_PEER_ID=peer2.org3.example.com
      - CORE_PEER_ADDRESS=peer2.org3.example.com:15051
      - CORE_PEER_LISTENADDRESS=0.0.0.0:15051
      - CORE_PEER_CHAINCODEADDRESS=peer2.org3.example.com:15052
      - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:15052
      - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer2.org3.example.com:15051
      - CORE_PEER_GOSSIP_BOOTSTRAP=peer0.org3.example.com:13051
      - CORE_PEER_LOCALMSPID=Org3MSP
    volumes:
        - /var/run/:/host/var/run/
        - ../crypto-
config/peerOrganizations/org3.example.com/peers/peer2.org3.example.com/msp:/etc/hyperl
edger/fabric/msp
        - ../crypto-
config/peerOrganizations/org3.example.com/peers/peer2.org3.example.com/tls:/etc/hyperl
edger/fabric/tls
        - peer2.org3.example.com:/var/hyperledger/production
    ports:
      - '15051'
      - '15053'
```

(2)Configure base/peer-base.yaml

```
gedit base/peer-base.yaml
```

```
# Copyright IBM Corp. All Rights Reserved.
#
# SPDX-License-Identifier: Apache-2.0
#

version: '2'

services:
  peer-base:
    image: hyperledger/fabric-peer
    environment:
      - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
      # the following setting starts chaincode containers on the same
      # bridge network as the peers
      # https://docs.docker.com/compose/networking/
```

```
      - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=${COMPOSE_PROJECT_NAME}_byfn
      - FABRIC_LOGGING_SPEC=INFO
      #- FABRIC_LOGGING_SPEC=DEBUG
      - CORE_PEER_TLS_ENABLED=true
      - CORE_PEER_GOSSIP_USELEADERELECTION=true
      - CORE_PEER_GOSSIP_ORGLEADER=false
      - CORE_PEER_PROFILE_ENABLED=true
      - CORE_PEER_TLS_CERT_FILE=/etc/hyperledger/fabric/tls/server.crt
      - CORE_PEER_TLS_KEY_FILE=/etc/hyperledger/fabric/tls/server.key
      - CORE_PEER_TLS_ROOTCERT_FILE=/etc/hyperledger/fabric/tls/ca.crt
      # Allow more time for chaincode container to build on install.
      - CORE_CHAINCODE_EXECUTETIMEOUT=300s
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric/peer
    command: peer node start

  orderer-base:
    image: hyperledger/fabric-orderer
    environment:
      - FABRIC_LOGGING_SPEC=INFO
      - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
      - ORDERER_GENERAL_BOOTSTRAPMETHOD=file
      - ORDERER_GENERAL_BOOTSTRAPFILE=/var/hyperledger/orderer/orderer.genesis.block
      - ORDERER_GENERAL_LOCALMSPID=OrdererMSP
      - ORDERER_GENERAL_LOCALMSPDIR=/var/hyperledger/orderer/msp
      # enabled TLS
      - ORDERER_GENERAL_TLS_ENABLED=true
      - ORDERER_GENERAL_TLS_PRIVATEKEY=/var/hyperledger/orderer/tls/server.key
      - ORDERER_GENERAL_TLS_CERTIFICATE=/var/hyperledger/orderer/tls/server.crt
      - ORDERER_GENERAL_TLS_ROOTCAS=[/var/hyperledger/orderer/tls/ca.crt]
      -
ORDERER_GENERAL_CLUSTER_CLIENTCERTIFICATE=/var/hyperledger/orderer/tls/server.crt
      -
ORDERER_GENERAL_CLUSTER_CLIENTPRIVATEKEY=/var/hyperledger/orderer/tls/server.key
      - ORDERER_GENERAL_CLUSTER_ROOTCAS=[/var/hyperledger/orderer/tls/ca.crt]
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric
    command: orderer
```

(3)Configure docker-compose-cli

```
gedit docker-compose-cli.yaml
```

```
# Copyright IBM Corp. All Rights Reserved.
#
# SPDX-License-Identifier: Apache-2.0
#

version: '2'

volumes: #order
  orderer.example.com:
  peer0.org1.example.com:
  peer1.org1.example.com:
  peer2.org1.example.com:
  peer0.org2.example.com:
  peer1.org2.example.com:
  peer2.org2.example.com:
  peer0.org3.example.com:
```

```
  peer1.org3.example.com:
  peer2.org3.example.com:

networks: #network
  byfn:

services:

  orderer.example.com:
    extends:
      file:   base/docker-compose-base.yaml
      service: orderer.example.com
    container_name: orderer.example.com
    networks:
      - byfn

  peer0.org1.example.com:
    container_name: peer0.org1.example.com
    extends:
      file:  base/docker-compose-base.yaml
      service: peer0.org1.example.com
    networks:
      - byfn

  peer1.org1.example.com:
    container_name: peer1.org1.example.com
    extends:
      file:  base/docker-compose-base.yaml
      service: peer1.org1.example.com
    networks:
      - byfn

  peer2.org1.example.com:
    container_name: peer2.org1.example.com
    extends:
      file:  base/docker-compose-base.yaml
      service: peer2.org1.example.com
    networks:
      - byfn

  peer0.org2.example.com:
    container_name: peer0.org2.example.com
    extends:
      file:  base/docker-compose-base.yaml
      service: peer0.org2.example.com
    networks:
      - byfn

  peer1.org2.example.com:
    container_name: peer1.org2.example.com
    extends:
      file:  base/docker-compose-base.yaml
      service: peer1.org2.example.com
    networks:
      - byfn

  peer2.org2.example.com:
    container_name: peer2.org2.example.com
    extends:
```

```
      file:  base/docker-compose-base.yaml
      service: peer2.org2.example.com
    networks:
      - byfn

  peer0.org3.example.com:
    container_name: peer0.org3.example.com
    extends:
      file:  base/docker-compose-base.yaml
      service: peer0.org3.example.com
    networks:
      - byfn

  peer1.org3.example.com:
    container_name: peer1.org3.example.com
    extends:
      file:  base/docker-compose-base.yaml
      service: peer1.org3.example.com
    networks:
      - byfn

  peer2.org3.example.com:
    container_name: peer2.org3.example.com
    extends:
      file:  base/docker-compose-base.yaml
      service: peer2.org3.example.com
    networks:
      - byfn


  cli:
    container_name: cli
    image: hyperledger/fabric-tools
    tty: true
    stdin_open: true
    environment:
      - GOPATH=/opt/gopath
      - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
      - FABRIC_LOGGING_SPEC=DEBUG #logfile
      #- FABRIC_LOGGING_SPEC=INFO
      - CORE_PEER_ID=cli
      - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
      - CORE_PEER_LOCALMSPID=Org1MSP
      - CORE_PEER_TLS_ENABLED=true
      -
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org1.example.com/peers/peer0.org1.example.com/tls/server.crt
      -
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org1.example.com/peers/peer0.org1.example.com/tls/server.key
      -
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt
      -
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org1.example.com/users/Admin@org1.example.com/msp
    working_dir: /opt/gopath/src/github.com/hyperledger/fabric/peer
    command: /bin/bash
    volumes:
```

```
        - /var/run/:/host/var/run/
        - ./../chaincode/:/opt/gopath/src/github.com/hyperledger/fabric-
samples/chaincode
        - ./crypto-config:/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
        - ./scripts:/opt/gopath/src/github.com/hyperledger/fabric/peer/scripts/
        - ./channel-
artifacts:/opt/gopath/src/github.com/hyperledger/fabric/peer/channel-artifacts
    depends_on:
      - orderer.example.com
      - peer0.org1.example.com
      - peer1.org1.example.com
      - peer2.org1.example.com
      - peer0.org2.example.com
      - peer1.org2.example.com
      - peer2.org2.example.com
      - peer0.org3.example.com
      - peer1.org3.example.com
      - peer2.org3.example.com
    networks:
      - byfn
```

4.Run docker-compose

```
docker-compose -f docker-compose-cli.yaml up
#or
docker-compose -f docker-compose-cli.yaml up -d
#Check running status
docker ps -a
```

```
root@admin:~/hyfa/network# docker ps -a
CONTAINER ID   IMAGE                      COMMAND             CREATED          STATUS         PORTS                                                                                  NAMES
5e19b04ad86f   hyperledger/fabric-tools   "/bin/bash"         37 seconds ago   Up 6 seconds                                                                                          cli
42a2ad29fc26   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 7 seconds   7051/tcp, 0.0.0.0:49349->13051/tcp, 0.0.0.0:49348->13053/tcp           peer0.org3.example.com
78cab2fa0f6f   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 7 seconds   7051/tcp, 0.0.0.0:49361->14051/tcp, 0.0.0.0:49359->14053/tcp           peer1.org3.example.com
0a804b8427c8   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 8 seconds   7051/tcp, 0.0.0.0:49345->9051/tcp, 0.0.0.0:49344->9053/tcp             peer2.org1.example.com
5e7975d4db5f   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 7 seconds   7051/tcp, 0.0.0.0:49352->15051/tcp, 0.0.0.0:49350->15053/tcp           peer2.org3.example.com
e2343da96005   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 7 seconds   7051/tcp, 0.0.0.0:49347->12051/tcp, 0.0.0.0:49346->12053/tcp           peer2.org2.example.com
147e779253b5   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 7 seconds   0.0.0.0:49362->7051/tcp, 0.0.0.0:49360->7053/tcp                        peer0.org1.example.com
58ce28e9af4e   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 7 seconds   7051/tcp, 0.0.0.0:49354->11051/tcp, 0.0.0.0:49351->11053/tcp           peer1.org2.example.com
ebcba1e239fa   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 7 seconds   7051/tcp, 0.0.0.0:49355->8051/tcp, 0.0.0.0:49353->8053/tcp             peer1.org1.example.com
37bd36e2a977   hyperledger/fabric-orderer "orderer"           39 seconds ago   Up 7 seconds   0.0.0.0:49358->7050/tcp                                                 orderer.example.com
cb01a31c0d43   hyperledger/fabric-peer    "peer node start"   39 seconds ago   Up 7 seconds   7051/tcp, 0.0.0.0:49357->10051/tcp, 0.0.0.0:49356->10053/tcp           peer0.org2.example.com
root@admin:~/hyfa/network#
```

2. Deploy smart contract

Copy fabric-simple abstore to network/chaincode/

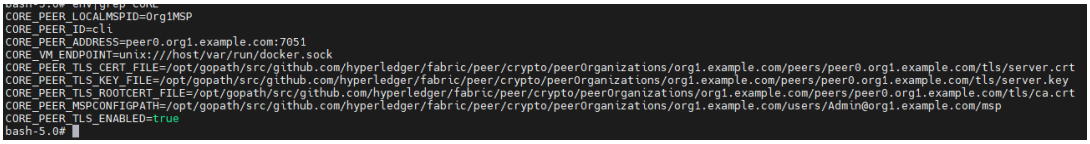
```
sudo cp -r /root/hyfa/fabric-samples/chaincode/abstore/
 /root/hyfa/network/chaincode
cd /root/hyfa/network/chaincode/abstore/go
go mod vendor
```

(1)Entering Docker container

```
docker exec -it cli bash
```

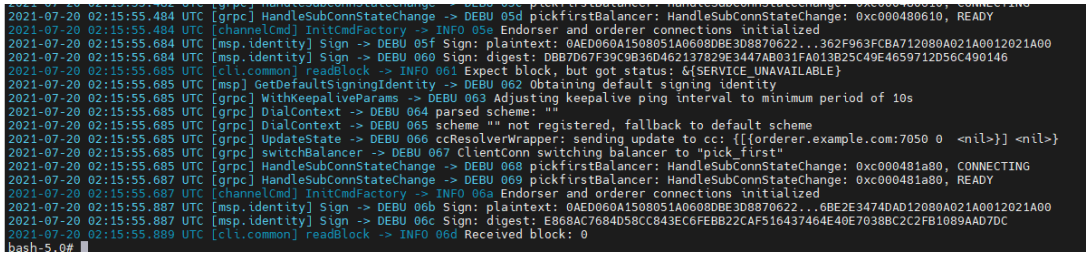
(2)Check peer information

```
env|grep CORE
```



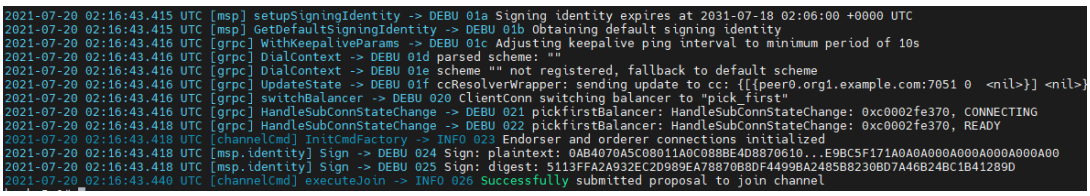
(3)Generate channel

```
export CHANNEL_NAME=mychannel
peer channel create -o orderer.example.com:7050 -c mychannel -f ./channel-
artifacts/mychannel.tx --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example
.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
```



(4)Joining channel

```
peer channel join -b mychannel.block
```

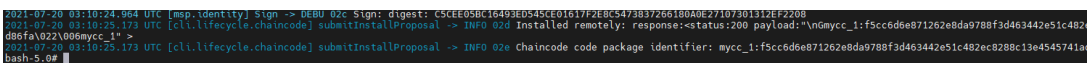


(5)Package and install smart contract and chaincode

```
peer lifecycle chaincode package mycc.tar.gz --path github.com/hyperledger/fabric-
samples/chaincode/abstore/go/ --lang golang --label mycc_1
```

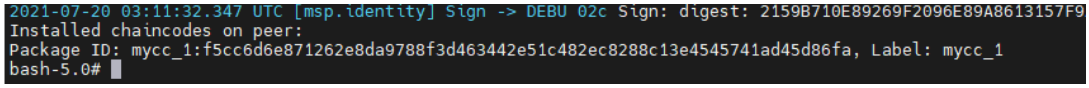
(6)Deploy smart contract to peer

```
peer lifecycle chaincode install mycc.tar.gz
```



(7) Check and save chaincode

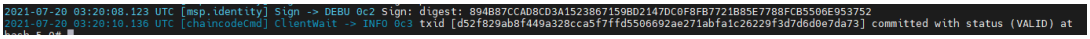
```
peer lifecycle chaincode queryinstalled
```




```
CC_PACKAGE_ID=mycc_1:f5cc6d6e871262e8da9788f3d463442e51c482ec8288c13e4545741ad45d86fa
```

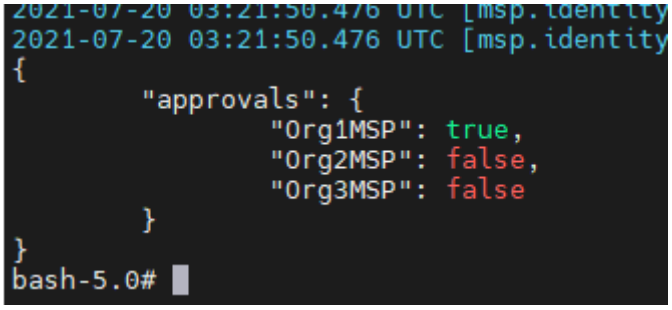
(8) Approve chaincode definition

```
peer lifecycle chaincode approveformyorg --channelID mychannel --name mycc --version
1.0 --init-required --package-id $CC_PACKAGE_ID --sequence 1 --tls true --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example
.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
```



(9)Committing the chaincode definition to the channel

```
peer lifecycle chaincode checkcommitreadiness --channelID mychannel --name mycc --
version 1.0 --sequence 1 --output json --init-required
```



Note: the current smart contract get the approve from org1MSP but haven't got approve from Org2MSP and Org3MSP.

The lifecycle strategy is to get more than half approve.

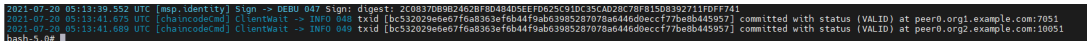
```
#   /Channel/Application/<PolicyName>
Policies:
    Readers:
        Type: ImplicitMeta
        Rule: "ANY Readers"
    Writers:
        Type: ImplicitMeta
        Rule: "ANY Writers"
    Admins:
        Type: ImplicitMeta
        Rule: "MAJORITY Admins"
    LifecycleEndorsement:
        Type: ImplicitMeta
        Rule: "MAJORITY Endorsement"
    Endorsement:
        Type: ImplicitMeta
        Rule: "MAJORITY Endorsement"
```

So switch to peer0.org2.example.com and peer0.org3.example.com and repeat the steps (4), (6)-(9) to get the following results.

```
2021-07-20 03:41:23.173 UTC [msp.identity]
2021-07-20 03:41:23.173 UTC [msp.identity]
{
        "approvals": {
                "Org1MSP": true,
                "Org2MSP": true,
                "Org3MSP": false
        }
}
bash-5.0#
```

(9)Submit a smart contract

```
peer lifecycle chaincode commit -o orderer.example.com:7050 --tls true --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example
.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem --
channelID mychannel --name mycc --peerAddresses peer0.org1.example.com:7051 --
tlsRootCertFiles
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.examp
le.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses
peer0.org2.example.com:10051 --tlsRootCertFiles
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.examp
le.com/peers/peer0.org2.example.com/tls/ca.crt --version 1.0 --sequence 1 --init-
required
```

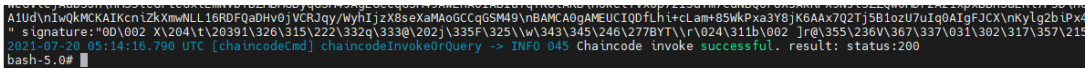


(10)Check the status of submitted smart cobtract

```
peer lifecycle chaincode querycommitted --channelID mychannel --name mycc
```

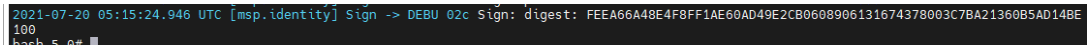
(11)Initialize smart contract

```
peer chaincode invoke -o orderer.example.com:7050 -C mychannel -n mycc --tls true --
cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example
.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem --
peerAddresses peer0.org1.example.com:7051 --tlsRootCertFiles
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.examp
le.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses
peer0.org2.example.com:10051 --tlsRootCertFiles
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.examp
le.com/peers/peer0.org2.example.com/tls/ca.crt --isInit -c '{"Args":
["Init","a","100","b","100"]}'
```



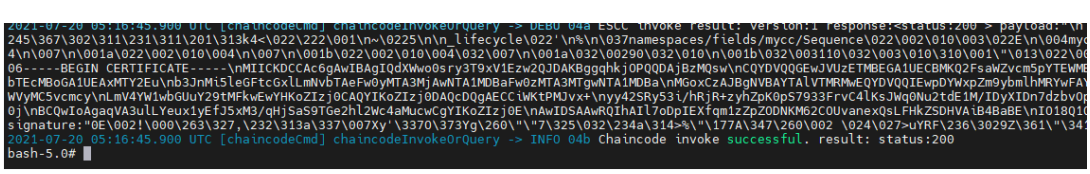
(12)Query

```
peer chaincode query -C mychannel -n mycc -c '{"Args":["query","a"]}'
```



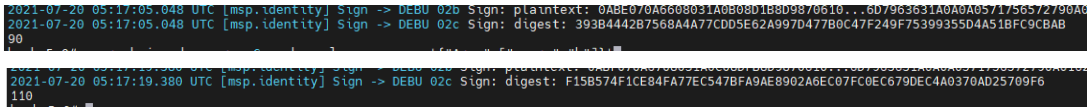
(13)Transfer

```
peer chaincode invoke -o orderer.example.com:7050 --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example
.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C
mychannel -n mycc --peerAddresses peer0.org1.example.com:7051 --tlsRootCertFiles
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.examp
le.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses
peer0.org2.example.com:10051 --tlsRootCertFiles
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.examp
le.com/peers/peer0.org2.example.com/tls/ca.crt -c '{"Args":["invoke","a","b","10"]}' -
-waitForEvent
```



(14)Query Again

```
peer chaincode query -C mychannel -n mycc -c '{"Args":["query","a"]}'
peer chaincode query -C mychannel -n mycc -c '{"Args":["query","b"]}'
```



Note: reputation will be calculated by averaging the previous reputation and new assigned contribution.

3. Modify the environment variable of each peer

(1)Org1

```
#Switch to peer0.org1.example.com
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer0.org1.example.com:7051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org1.example.com/peers/peer0.org1.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org1.example.com/peers/peer0.org1.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org1.example.com/users/Admin@org1.example.com/msp
CORE_PEER_TLS_ENABLED=true
#Switch to peer1.org1.example.com
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer1.org1.example.com:8051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org1.example.com/peers/peer1.org1.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org1.example.com/peers/peer1.org1.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org1.example.com/peers/peer1.org1.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org1.example.com/users/Admin@org1.example.com/msp
CORE_PEER_TLS_ENABLED=true
#Switch to peer2.org1.example.com
CORE_PEER_LOCALMSPID=Org1MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer2.org1.example.com:9051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org1.example.com/peers/peer2.org1.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org1.example.com/peers/peer2.org1.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org1.example.com/peers/peer2.org1.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org1.example.com/users/Admin@org1.example.com/msp
```

```
CORE_PEER_TLS_ENABLED=true
```

(2)Org2

```
#Switch to peer0.org2.example.com
CORE_PEER_LOCALMSPID=Org2MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer0.org2.example.com:10051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org2.example.com/peers/peer0.org2.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org2.example.com/peers/peer0.org2.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org2.example.com/users/Admin@org2.example.com/msp
CORE_PEER_TLS_ENABLED=true
#Switch to peer1.org2.example.com
CORE_PEER_LOCALMSPID=Org2MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer1.org2.example.com:10051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org2.example.com/peers/peer1.org2.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org2.example.com/peers/peer1.org2.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org2.example.com/peers/peer1.org2.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org2.example.com/users/Admin@org2.example.com/msp
CORE_PEER_TLS_ENABLED=true
#Switch to peer2.org2.example.com
CORE_PEER_LOCALMSPID=Org2MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer2.org2.example.com:10051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org2.example.com/peers/peer2.org2.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org2.example.com/peers/peer2.org2.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org2.example.com/peers/peer2.org2.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org2.example.com/users/Admin@org2.example.com/msp
CORE_PEER_TLS_ENABLED=true
```

(3)Org3

```
#Switch to peer0.org3.example.com
CORE_PEER_LOCALMSPID=Org3MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer0.org3.example.com:13051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org3.example.com/peers/peer0.org3.example.com/tls/server.crt
```

```
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org3.example.com/peers/peer0.org3.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org3.example.com/peers/peer0.org3.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org3.example.com/users/Admin@org3.example.com/msp
CORE_PEER_TLS_ENABLED=true
#Switch to peer1.org3.example.com
CORE_PEER_LOCALMSPID=Org3MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer1.org3.example.com:13051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org3.example.com/peers/peer1.org3.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org3.example.com/peers/peer1.org3.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org3.example.com/peers/peer1.org3.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org3.example.com/users/Admin@org3.example.com/msp
CORE_PEER_TLS_ENABLED=true
#Switch to peer2.org3.example.com
CORE_PEER_LOCALMSPID=Org3MSP
CORE_PEER_ID=cli
CORE_PEER_ADDRESS=peer2.org3.example.com:13051
CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
CORE_PEER_TLS_CERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org3.example.com/peers/peer2.org3.example.com/tls/server.crt
CORE_PEER_TLS_KEY_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerO
rganizations/org3.example.com/peers/peer2.org3.example.com/tls/server.key
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/
peerOrganizations/org3.example.com/peers/peer2.org3.example.com/tls/ca.crt
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peer
Organizations/org3.example.com/users/Admin@org3.example.com/msp
CORE_PEER_TLS_ENABLED=true
```

Delete Volume

```
docker-compose -f docker-compose-cli.yaml down --volumes --remove-orphans
docker rm -f $(docker ps -a | grep "hyperledger/*" | awk "{print \$1}")
docker volume prune
```

Appendix D. Chapter 6 Appendix

D.1 The ground truth of case studies

The experimental three case studies with ground truth are shown in Table D.1. And the simulated responses from five experts are shown in Table D.2.

Table D.1: The ground truth of case studies

ID	Request	Ground Truth
	Case 1: Car Service	
1	When should I service my car?	Every 6 months or 10,000km whichever comes first
2	How often should you check your oil?	Better to check them every week
3	How often should you check your tyre pressure?	You should check your tyre pressure once a month.
4	What is Third Party Car Insurance?	Third Party Car Insurance offers cover for damage caused to someone else's vehicle or property, if you're liable for it. It can also covers for loss of, or damage caused by fire or theft.
5	What's covered by Third Party Car Insurance?	Third Party Property Damage Car Insurance covers damage you cause to other people's vehicles and property while behind the wheel.
6	What is Comprehensive Car Insurance?	Comprehensive Car Insurance is a type of car insurance that covers you for: 1) accidental damage to your car; 2) damage that may be caused by the use of your car cause to other vehicles and property; 3) theft, fire, and malicious damage to your car.
7	What's the difference between Comprehensive and Third Party Car Insurance?	Comprehensive will provide you with coverage for a range of insured events such as accidents and theft, as well as weather events like hail, fire and storms. If you're involved in an accident, your car and property and the other person's car and property are covered. Third Party Car Insurance will only cover you for damage you cause to someone else's vehicle and property.

8	What's a flexi excess?	Flexi excess is an additional excess payment you can choose, on top of your standard excess.
9	Does Comprehensive Car Insurance cover fire and theft?	Yes, Comprehensive Car Insurance covers you for accidental loss or damage in the event of fire, theft or attempted theft.
10	Does Comprehensive Car Insurance cover windscreen replacement?	Yes, Comprehensive Car Insurance does cover windscreen replacement as standard, but you will need to pay an excess if you claim.
11	Can I add on Roadside Assist?	Yes, Roadside Assist can be added to any Comprehensive Car Insurance policy.
12	Is CTP insurance included in registration?	No, they're not one and the same payment, as they are only in some states.
13	How is CTP Insurance calculated?	When insurers calculate CTP Insurance, they'll consider the cost of future claims, as well as a number of other factors like age, safety record, demerit points and claims history.
14	What is MAI Insurance?	MAI Insurance covers everybody who is injured in a motor vehicle accident.
15	What happens to my MAI Insurance if I buy or sell a vehicle that is registered?	MAI Insurance is linked to the vehicle, not the owner. If you sell your vehicle, the MAI Insurance will be transferred to the new owner. Similarly, if you buy a vehicle, the MAI Insurance will be transferred to you.
16	How can I update my contact information on my MAI Insurance policy?	Since your MAI Insurance is connected to your vehicle registration, you should contact Access Canberra to update your contact information.
17	How is car insurance calculated?	Many factors can influence how much your car insurance premium will cost, including: how expensive your car is, and how expensive replacement parts would be the age of your car, certain safety features where you live, and the age, experience, and driver history of the insured driver.
18	Does young driver have more expensive car insurance?	Yes, insurers take into consideration possible risks. Young drivers may be more inexperienced on the road, and therefore are sometimes more likely to have an accident. For this reason, car insurance can end up costing more for a young driver.
19	If I had a car accident, how to use my car insurance?	If an event happens that's covered by your car insurance, you may need to make a claim. You may have to pay an excess (particularly if you're at fault).
20	How much can I insure my car for?	You can choose the amount you want to cover your car for based on details about your car and you, will give you a minimum and maximum amount you can cover your car for. Then you can adjust the amount to any number within that range, and that will become the maximum amount.
Case 2: English Learning		

21	what does "Ace" mean?	If something is ace it is awesome. Kids thought all cool stuff was ace, or brill.
22	What does "Not my cup of tea" mean?	This is a common saying that means something is not to your liking.
23	What does "Spend a penny" mean?	To spend a penny is to go to the bathroom. It comes from the fact that ladies used to operate the door by inserting an old penny.
24	What does "Wind up " mean?	This has a couple of meanings. If something you do is a "wind up" it means you are making fun of someone. However it you are "wound up" it means you are annoyed.
25	what is "sixes and sevens"?	If something is all at sixes and sevens then it is in a mess, topsy turvy or somewhat haywire!
26	What does "DJ" mean?	It means Dinner jacket. We usually refer to it as our DJ. Sometimes it also means Disc Jockey.
27	What is "Swimming costume"?	This is what you wear to go swimming. You might call it a bathing suit. We also say swimsuit and cozzy.
28	what does "Bin liner" mean?	This is another word for bin bag.
29	What is des res?	If someone lives in a particularly nice property in a nice part of town it would be referred to as a des res. It is short for desirable residence and usually means bloody expensive.
30	What is "White goods"?	These are the electrical appliances that you have in your kitchen or utility room like fridges, freezers, washing machines and driers. The name is cunningly derived from their colour.
31	What is "Bugger all"?	If something costs bugger all, it means that it costs nothing. Meaning it is cheap. If you have bugger all, it means you have nothing.
32	What is "Dog's dinner"?	If you make a real mess of something it might be described as a real dog's dinner.
33	What does Donkey's years mean?	Someone said to me the other day that they hadn't seen me for donkey's years. It means they hadn't seen me for ages.
34	What does "Good value" mean?	This is short for good value for money. It means something is a good deal.
35	What does "Horses for courses" mean?	This is a common saying that means each to his own. What suits one person might be horrible for someone else.
36	What does "I'm easy" mean?	This expression means I don't care or it's all the same to me.
37	If someone have knees up, means what?	If you're having a knees up, you're going to a dance or party, generally having a great time. Usually involving alcohol!
38	"Looking left, right and centre " means what?	If you have been looking left, right and centre, it means you have been searching all over.
39	What are you on about? means what?	It means what are you talking about?
40	If someone said "Put a sock in it ", it means what?	This is one way of telling someone to shut up

41	When I visited British, I heard "Best of British" , what does this mean?	If someone says "The best of British to you" when you are visiting the UK, it simply means good luck. It is short for "best of British luck".
42	Something is a blinding success, means what?	It means it was awesome.
43	"Box your ears" means what?	Generally meant a slap around the head for misbehaving.
44	what does "Brassed off" mean?	If you are brassed off with something or someone, you are fed up.
45	To Flog the old TV means what?	To Flog something means to sell it.
46	What does "Gen" mean?	Gen means information. If you have the gen then you know what is going on.
47	She is good at haggling means what?	To haggle is to argue or negotiate over a price.
48	Kip means what?	A short sleep or a snooze.
49	If something has gone pear shaped, what does that mean?	It means it has become a disaster.
50	If someone is zonked, it means what?	It means they are totally knackered or you might say exhausted.
Case 3: Health & Safety Training		
51	What's your duty of care if you are a officer?	It includes: 1) acquire safety knowledge and keep up to date; 2) understand operations and associated risks; 3) ensure WHS Legal Compliance; 4) receive and consider information on incidents, hazards and risks; 5) ensure resources and process to eliminate or minimise risks; 6) verify the provision and use of WHS resources
52	What's your duty of care if I am a student?	Take reasonable care for your own safety and the safety of others; Comply with any reasonable instruction, policy or procedures of the University in relation to work health and safety; Report all hazards and incidents to your Manager/Supervisor as soon as possible.
53	Who will review the safety and wellbeing performance every meeting ?	University Council will review the safety and wellbeing performance every meeting
54	What is a safe system of work?	A safe system of work is a way of doing things safely at the University. It's about incorporating safety into our day to day work and decisions.
55	What elements include in safe system of work?	It includes commitment, consultation, safe Work Procedures, Training and Supervision, Reporting Safety and Continuous Improvement, Injury Management and Return to Work.
56	What will prioritise your time and budget to meet safety responsibilities?	Commitment

57	How can you demonstrate you follow Safe Work Procedures?	You can use the procedures that apply to your activities, or raise improvements you identify to do the activity safer or more effectively.
58	How can you demonstrate you report on safety?	Use MySAFETY for reporting safety issues and incidents or Use PocketSafety App when out and about.
59	How to get SafeZone app?	Follow the download instructions on the SafeZone webpage. And once you downloaded, sign up for campuses relevant to your work
60	What type of incidents need to report in MySAFETY?	All incidents must be reported in MySAFETY

D.2 Simulated responses and fusion results

Table D.2: The simulated responses and fusion results

ID	**Response 1**	**Response 2**	**Response 3**	**Response 4**	**Response 5**	**Fusion result**
1	every 6 months	every 10,000km	Normally half an year or every 10,000km	normally every 20,000km	every 10,000km	normally every 6 months or every 10,000km
2	every three months	every week will be better	It is better to check every week.	every month	every week	every week will be better.
3	every three months	once a month	Better to check every month	every month	every two months	better to check every month.
4	Third Party Car Insurance is a type of car insurance covers for loss or damage caused by someone else's vehicle	It is a type of car insurance covers for damage caused by someone else's vehicle or property if you are responsible.	Third Party Car Insurance covers for damage caused to someone else's vehicle and for loss of, or damage by fire or theft.	It is a type of car insurance covers for loss or damage caused by someone else's vehicle or property if you are liable for it.	Third Party Car Insurance covers for damage caused to someone else's vehicle or property, and also covers for loss of, or damage by fire or theft.	Third Party Car Insurance is a type of car insurance covers for damage caused by someone else's vehicle or property if you are responsible, also covers for loss of, or damage by fire or theft.
5	It covers damage caused by someone else's vehicles or property.	Third Party Property Car Insurance covers for damage caused by other people's vehicles and property if you are liable for it.	It covers damage caused by other people's vehicles or property	Third Party Property Car Insurance covers for damage caused by someone else's vehicles or property while behind the wheel.	Third Party Car Insurance covers for damage caused to other people's vehicle and property.	Third Party Car Insurance is a type of car insurance covers for damage caused by someone else's vehicle or property while behind the wheel.

185

Table D.2 continued from previous page

6	Comprehensive Car Insurance is a type of car insurance that covers for accidental damage to your car.	Comprehensive Car Insurance covers for accidental damage to your car or damage caused by theft or fire.	It covers damage caused by other people's vehicles or property, accidental damage, fire, theft and other malicious damage.	It covers for loss or damage caused by someone else's vehicle or property, or damage caused by theft, fire or other malicious damage.	It covers damages caused by accidents, theft or fire.	Comprehensive Car Insurance covers for accidental damage to your car, damage caused by theft or fire, or other people's vehicles or property or other malicious damage.
7	Comprehensive Car Insurance will cover accidents, theft, fire, storms and also other people's vehicle or property. Third Party Car insurance will only cover for damage caused by other people's vehicle and property.	Comprehensive Car Insurance covers for accidental damage to your car or damage caused by theft or fire. While third Party Car Insurance only covers for damage caused by someone else's vehicle and property.	Comprehensive car Insurance covers for damages caused by someone else or cause by your fault but third party car insurance only covers the damages caused by other people.	Comprehensive car Insurance covers a lot but third party car insurance only covers the damages caused by other people.	Third party car insurance only covers the damages caused by other people's car or property and comprehensives one covers more wide range.	Comprehensive Car Insurance covers for damages caused by other people's vehicle and property, accidents, theft, fire or caused by your fault. Third Party car insurance only covers the damages caused by other people.

Table D.2 continued from previous page

8	It is an extra payment that you can choose.	It is extra payment on top of your standard excess.	It is an additional payment that you can select.	Flexi excess is an extra excess payment on the top of your standard excess.	Flexi excess is some extra payment you can choose by yourself.	Flexi excess is an extra excess payment that you can choose on top of your standard excess.
9	Yes, it covers you for fire and theft.	Yes, comprehensive car insurance covers accidental damages, or fire, theft.	Yes, it covers you for accidental damages, loss, fire and theft.	No, it only covers accidental damages.	Yes, all covered.	Yes, comprehensive car insurance covers you for accidental damages, fire and theft.
10	No, doesn't cover and need an excess	Yes, it covers windscreen replacement as standard but you may need extra payment if you claim.	Yes, it covers windscreen replacement	Yes, it covers windscreen replacement	No, you need to pay an excess if you claim	Yes, it covers windscreen replacement as standard but need excess payment if you claim
11	Yes you can.	Yes, roadside assist can be added to your comprehensive car insurance.	Yes, it can be added to your car insurance policy.	No, you need extra excess payment	Not sure, better to contact your insurance company.	Yes, roadside assist can be added to your comprehensive car insurance policy.
12	Yes, it is included in registration	No, it is not included.	No, CTP insurance is not included because it is not in every states.	No, CTP insurance is not included since only some states have CTP insurance.	Yes, it is included in some states.	No, CTP insurance is not included because it is not in some states.

Table D.2 continued from previous page

13	It will consider the cost of claims in the future and claim history.	The insurers will consider the cost of future claims, claim history and safety record.	The insurers will calculate considering the cost of future claims, car age, demerit points, and others.	It will be calculated based on claim history, car age as well as future claims.	They may consider the claim history, car age and demerit points.	The insurers will consider the cost of future claims, claim history, safety record, car age, demerit points and others.
14	MAI covers people who is injured in a accident when they drive motor vehicle.	MAI is a type of insurance that covers everybody injured in a motor vehicle accident.	It is an insurance that covers everybody who is injured in a motor vehicle accident.	It is an insurance when motor vehicle accident happened.	It is for people who is injured in a motor vehicle accident.	MAI is a type of insurance for people who is injured in a motor vehicle accident.
15	If you buy or sell a vehicle, MAI belongs to car owner.	If you buy a vehicle, MAI will be transferred to you, and if you sell your vehicle, the MAI will be transferred to the new owner.	If you buy a vehicle, MAI will be transferred to you, while if you sell your vehicle, the MAI will be transferred to the new owner of the vehicle.	No matter you buy or sell a vehicle, MAI can be always transferred to you.	MAI is registered in the vehicle, so if you buy a new car, it will be transferred to you, and if you sell your car, the MAI will be transferred to the new owner.	MAI is registered in the vehicle. If you buy a vehicle, MAI will be transferred to you, and if you sell your vehicle, the MAI will be transferred to the new owner.
16	You should contact your insurance company	You should contact access Canberra to update.	You should contact Access Canberra to update your contact information	You should contact Access Canberra to update and it is because MAI is linked to your vehicle registration.	You should contact your insurance company to update your contact information	You should contact Access Canberra to update your contact information and it is because MAI is linked to your vehicle registration.

Table D.2 continued from previous page

17	It is calculated based many factors such as price of your car, car age, experience, driving history, etc.	Many factors are considered to calculate, including the price of your car, how expensive the replacement parts would be, the age of the car, insured history, etc.	Many factors will be considered such as how expensive your car is, how expensive to replace parts of your car, the car age, driver history, etc.	it is calculated based on price of the car, car age, driver history, etc.	Many factors including car age, car price, driver history, and so on.	Many factors are considered to calculate including price of your car, car age, driver history, how expensive the replacement parts would be, experience, etc.
18	Yes, because they have less experienced, may cause more accidents	Yes, the car insurance will be calculated considering the driver experience, young driver will pay more car insurance.	Yes, because young driver may be less experienced on the road, and more likely to have accident. So younger driver will have more expensive car insurance.	Yes	Yes, car insurance can be more for young driver, because they are more likely to have accident.	Yes, car insurance will be calculated considering the driver experience, because young driver may have less experiences, they are more likely to have accident. Young driver will have more expensive car insurance.
19	You need to make a claim.	If it is covered in your car insurance, you can make a claim	If it is covered by your car insurance, you need to make a claim, and sometimes you may need to pay an excess.	You need to make a claim, and if it is your fault, you may need to pay an excess.	Make a claim first, and if it is your fault, you may need to pay an excess.	If it is covered by your car insurance, you need to make a claim first, and if it is your fault, you may need to pay an excess.

Table D.2 continued from previous page

20	You can choose amount covered, we will give a range based on the information of your car and you. Then you can change to any number, and that will be the maximum amount.	We will give a minimum and maximum amount you can cover your car for based on provided car details and your information, then you can adjust to any number that you like.	You can choose the amount you would like to cover, including minimum and maximum amount.	The minimum and maximum amount will be recommended based on the details of your car and you, then you can adjust to any number you like.	The amount covered is decided by you.	You can choose amount covered. We will give a minimum and maximum amount based on the details of your car and you, and you can adjust to any number that you like.
21	Ace means awesome	Ace means something awesome.	Something is ace means it is brill.	people think cool stuff are ace.	awecome	ace means something awesome, brill, people think cool stuff are ace.
22	It means something is not to your liking.	If something is not your cup of tea, then that means you do not like it.	Means something you do not like it.	It means you don't like it.	Something that a person finds to be not agreeable.	If something is not your cup of tea, it means you do not like it.
23	It means someone would like go to toilet.	To spend a penny means go to the toilet.	It means go to the bathroom.	It means someone operate the door by inserting a penny.	Means to go to the toilet.	To spend a penny means go to the toilet.

Table D.2 continued from previous page

24	It means you are annoyed.	it means you are making fun or sometimes also means you are annoyed.	It means you are making fun of someone.	It means you are making fun or being silly.	It means you are annoyed.	It means you are making fun or being silly, also means you are annoyed.
25	Means mess	if something is at sixes and sevens, then it is mess.	means mess or haywire.	It means a bit mess or topsy turvy.	means people are all six and seven years' old.	If something is at sixes and sevens, it means mess or haywire or topsy turvy.
26	Means dinner jacket.	It means tuxedo.	We usually refer to Dinner jacket as our DJ	Disc Jockey	means Disc Jockey	DJ means Dinner jacket or Disc Jockey.
27	This is what you wear to swim.	It is your bath suit.	It is what you need to wear during swimming	It is what you wear when you swim, also call cozzy.	We also call bath suit or swim suit.	It is what you wear to swim, also call bath suit or cozzy.
28	Means bin bag.	It is the black bag that you put inside the kitchen bin.	Bin bag	Means bin bag that you put the kitchen bin inside.	It is the black bag that you put inside the bin.	Bin liner means bin bag that you put the kitchen bin inside.
29	Usually means desirable residence	It means someone lives in a nice property in a nice place.	It is short for desirable residence.	Means a very desirable residence	Means someone lives a nice house or apartment.	Des res is short for desirable residence that means someone lives in a nice property in a nice place.

Table D.2 continued from previous page

30	Means electrical appliances with white colours.	They are electrical appliances that in the kitchen, such as fridges, washing machines, driers, etc.	They are electrical appliances with write colours, like fridges, freezers, washing machine.	They means electrical appliances in the kitchen or utility room.	Means all the appliances with white colours.	White goods are electrical appliances with white colours that in the kitchen, such as fridges, freezers, washing machine, driers, etc.
31	It means it costs nothing.	It means it is free.	It means it is cheap.	If you have bugger all, it means you have nothing.	It means very little or nothing.	Bugger all means it costs nothing or very cheap.
32	It means a very mess dinner	It means you make a dinner for your dog.	It means you make a very mess of something	It means something you make really mess.	It means you make something really mess.	Dog's dinner means you make a very mess of something.
33	It means someone hasn't seen me for a long time	It means someone hasn't seen me for ages.	If someone said they hadn't seen me for donkey's years, it means they hadn't seen me for a long time.	Means someone hadn't seen me for many years.	It means two people haven't seen each other for a long time.	If someone said they hadn't seen me for donkey's years, it means someone hasn't seen me for a long time.
34	It is short for good value for money	It means something is a good deal.	It means something has a good value for money.	It means something is worth buying	means it is a good deal.	It is short for good value for money, and it means something is a good deal.

Table D.2 continued from previous page

35	It means something suits one person might not be suitable for someone else.	It means what suits one person might be horrible for other people.	It means each to his own.	It means each to his own, and something suits you might not be suitable for others.	It means something you like while others dislike.	It means something suits one person might not be suitable for someone else, means each to his own.
36	Means something is easy for me.	Means I don't care	means it's all the same to me	Means I don't care or it is all the same to me.	Means nothing important for me.	Means I don't care or it is all the same to me.
37	Means someone will go a party.	Means someone will go to dance or party.	It means you are going to a party and having a good time.	It means someone are going to a party and enjoying alcohol.	It means someone are going to a dance or party	It means someone will go a party or dance and having a good time and enjoying alcohol.
38	It means looking around.	It means you are searching something all over.	It means something you search all over.	It means something are have been searching all over.	It means you are looking left, right and center.	It means you are searching something all over.
39	It means what are you doing?	It means what are you thinking?	It means what are you talking about?	It means what are you talking about?	It means what do you mean?	It means what are you talking about?
40	Means shut up	Means to stop talking;	It is a way to shut someone's mouth	Means wear your socks	Means shut up	It means shut up.
41	It means good luck.	This is short for best of British luck.	It means good luck and it is short for best of British luck.	It means good luck	Means good bye.	It means good luck and it is short for best of British luck.

Table D.2 continued from previous page

42	It means someone blind your eyes.	It means something is awesome.	Means turning a blind eye	Meas awesome	It means something is really awesome.	It means something is awesome.
43	It means a slap around the ear	It generally means a slap around the head.	It means hit across the side of the head.	It means a punch around the head.	It is to slap someone upside the head.	It generally means a slap around the head.
44	It means you are fed up with something or someone.	If something or someone you are brassed off, it means to be fed up.	It means you are fed up with something or someone.	It means something make you fed up.	Means to be very much fed up	It means you are fed up with something or someone.
45	Means you would like to sell this old TV.	Means the old TV is going to sell.	To Flog the old TV means to sell it.	It means to beat it.	It means to beat the old TV with a whip.	To Flog the old TV means to sell this old TV.
46	It means information	It means knowledge	It means information or knowledge.	It means information	It means summary of key points.	It means information or knowledge.
47	Means she is good at arguing over a price.	means she is good at arguing over the terms of a purchase	It means she is good at arguing or negotiating over a price	Means she is good at arguing or negotiating over a price or agreement	Means bargaining	It means she is good at arguing or negotiating over a price or agreement.
48	Means a nap	Means a short sleep	Means a short nap	Means a snooze	Means a short sleep	Means a short nap or a snooze.

Table D.2 continued from previous page

49	It means something has become bad	It means something has become a disaster	It means something has been more negative	It means something has become a disaster	It means the shape of pear is gone.	It means something has become a disaster
50	Please turn off	It is short for "please turn off"	It means please turn off the form	Please turn off	Please turn off	It is short for please turn off the form.
51	It includes acquiring safety knowledge and keep up to date.	It includes understanding operations and associated risks	It includes receiving and considering information on incidents, hazards and risks	It includes verify the provision and use of WHS resources	It includes ensuring resources and process to eliminate or minimize risks.	It includes acquiring safety knowledge and keep up to date, and understanding operations and associated risks, and receiving and considering information on incidents, hazards and risks, and verify the provision and use of WHS resources, and ensuring resources and process to eliminate or minimize risks.

Table D.2 continued from previous page

52	Take care of your own safety.	Take care of your own safety, an report any hazards and incidents to your manager asap.	It includes taking care of your safety and the safety of others.	It includes comply with any respsonale instruction, policy and procedures of the unitiverty related to work health and safety.	It includes taking care of your safety and report all hazards, incidents, risks to your supervisor.	It includes taking care of your own safety and the safety of others, report any hazards and incidents, risks to your manager or supervisor asap, and comply with any reseasonale instruction, policy and procedures of the unitiverty related to work health and safety
53	University Council	University Executive Team	Safety and Wellbeing Team	University Council	University Council	University Council
54	It is a way to make sure doing things safety	It is about incorporating safety into our daily work at the university	It is a way to make sure doing our daily work safety	It is a way of doing our day to day work and decisions safety.	It is a safety system to make sure doing things safety at the university.	It is a way to make sure doing our daily work and decisions safety at the university
55	It includes commitment, consultation, safe work procedures, etc.	It includes commitment, consultation, training and supervision, reporting safety.	It includes training and supervision, reporting safety and continous improvement, injury management and back to work.	It includes commitment, consultation, supervision, reporting safety and continous improvement.	It includes commitment, reporting safety, injury management and return to work, etc.	It includes commitment, consultation, safe work procedures, training and supervision, reporting safety and continous improvement, injury management and back to work.

Table D.2 continued from previous page

56	Commitment	Commitment or Consultation	Commitment	Commitment and Consultation	Commitment	Commitment
57	You can apply the safe work procedures to your activities.	You can use safe work procedures that apply to your activities.	You can make improvements when you identify to do the activity safer.	You can improve your activity to make it safer or more effectively.	You can use the procedures to apply your activities to make it safer or more effectively.	You can use the safe work procedures to your activities, and make improvements safer or more effectively.
58	Use MySAFETY app to report	Use MySAFETY to report safety issues and incidents	Use PocketSafety App to report when you are out	Use MySAFETY or PocketSafety	Use either MySAFETY or PocketSafety	Use MySAFETY app to report safety issues and incidents or use PocketSafety App to report when you are out.
59	You can download on the SafeZone web page.	Please follow the download instruction on the webpage	You can follow the instruction on the SafeZone webpage, and need sign up for campuses relevant to your work	You can download on the webpage, and when you download, sign up for campuses related to your work.	Follow the download instruction on the SafeZone webpage	You can follow the download instruction on the SafeZone webpage, and need sign up for campuses relevant to your work.
60	Serious incidents must be reported.	Dangerous incidents need to be reported.	Serious injury need to report.	All the incidents need to report.	Only serious incidents need to report.	All the incidents must be reported.

Bibliography

(n.d.).

URL: *https://www.ahgautoservice.com.au/blog/a-brief-car-service-guide*

Abraham, I., Malkhi, D. et al. (2017), 'The blockchain consensus layer and bft', *Bulletin of EATCS* **3**(123).

Ahmad, A., Saad, M., Njilla, L., Kamhoua, C., Bassiouni, M. & Mohaisen, A. (2019), Blocktrail: A scalable multichain solution for blockchain-based audit trails, *in* 'ICC 2019-2019 IEEE International Conference on Communications (ICC)', IEEE, pp. 1–6.

Aldakheel, J. S., AlAhmad, M. A. & Al-Foudery, A. (2019), 'Comparison between bitcoin and quarkchain', *Journal of Computational and Theoretical Nanoscience* **16**(3), 818–822.

Aldeen, Y. A. A. S., Salleh, M. & Razzaque, M. A. (2015), 'A comprehensive review on privacy preserving data mining', *SpringerPlus* **4**(1), 1–36.

Ali, M., Nelson, J. C., Shea, R. & Freedman, M. J. (2016), Blockstack: A global naming and storage system secured by blockchains., *in* 'USENIX Annual Technical Conference', pp. 181–194.

Alla, H. B., Alla, S. B., Touhafi, A. & Ezzati, A. (2018), 'A novel task scheduling approach based on dynamic queues and hybrid meta-heuristic algorithms for cloud computing environment', *Cluster Computing* **21**(4), 1797–1820.

Allcott, H. & Gentzkow, M. (2017), 'Social media and fake news in the 2016 election', *Journal of Economic Perspectives* **31**(2), 211–36.

Aniello, L., Baldoni, R., Gaetani, E., Lombardi, F., Margheri, A. & Sassone, V. (2017), A prototype evaluation of a tamper-resistant high performance blockchain-based transaction log for a distributed database, *in* 'Dependable Computing Conference (EDCC), 2017 13th European', IEEE, pp. 151–154.

Ardor White-paper (2017).

URL: *https://ardordocs.jelurida.com/Getting*$_s tarted$

Arora, S., Batra, K. & Singh, S. (2013), 'Dialogue system: A brief review', *arXiv preprint arXiv:1306.4134* .

Atzei, N., Bartoletti, M. & Cimoli, T. (2017), A survey of attacks on ethereum smart contracts (sok), *in* 'Principles of Security and Trust', Springer, Berlin, Heidelberg, pp. 164–186.

Aydin, S. & Kahraman, C. (2013), 'A new fuzzy analytic hierarchy process and its application to vendor selection problem.', *Journal of Multiple-Valued Logic & Soft Computing* **20**.

Azbeg, K., Ouchetto, O., Andaloussi, S. J. & Fetjah, L. (2021), 'An overview of blockchain consensus algorithms: Comparison, challenges and future directions', *Advances on Smart and Soft Computing* pp. 357–369.

Bach, L. M., Mihaljevic, B. & Zagar, M. (2018), Comparative analysis of blockchain consensus algorithms, *in* '2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)', IEEE, pp. 1545–1550.

Back, A., Corallo, M., Dashjr, L., Friedenbach, M., Maxwell, G., Miller, A., Poelstra, A., Timón, J. & Wuille, P. (2014), 'Enabling blockchain innovations with pegged sidechains', *URL: http://www. opensciencereview. com/papers/123/enablingblockchain-innovations-with-pegged-sidechains* .

Barais, O., Le Meur, A. F., Duchien, L. & Lawall, J. (2008), Software architecture evolution, *in* 'Software Evolution', Springer, Berlin, Heidelberg, pp. 233–262.

Barkai, D. (2000), 'An introduction to peer-to-peer computing', *Intel Developer update magazine* pp. 1–7.

Barrera, D., Chuat, L., Perrig, A., Reischuk, R. M. & Szalachowski, P. (2017), 'The scion internet architecture', *Communications of the ACM* **60**(6), 56–65.

Bashir, I. (2017), *Mastering blockchain*, Packt Publishing Ltd.

Bertino, E., Khan, L. R., Sandhu, R. & Thuraisingham, B. (2006), 'Secure knowledge management: confidentiality, trust, and privacy', *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and humans* **36**(3), 429–438.

Bhargavan, K., Delignat-Lavaud, A., Fournet, C., Gollamudi, A., Gonthier, G., Kobeissi, N., Kulatova, N., Rastogi, A., Sibut-Pinote, T., Swamy, N. et al. (2016), Formal verification of smart contracts: Short paper, *in* 'Proceedings of the 2016 ACM Workshop on Programming Languages and Analysis for Security', ACM, pp. 91–96.

Birman, K., Malkhi, D. & Van Renesse, R. (2010), 'Virtually synchronous methodology for dynamic service replication'.

Bissias, G., Levine, B. N., Ozisik, A. P. & Andresen, G. (2016), 'An analysis of attacks on blockchain consensus', *arXiv preprint arXiv:1610.07985* .

Black, E. & Hunter, A. (2009), 'An inquiry dialogue system', *Autonomous Agents and Multi-Agent Systems* **19**(2), 173–209.

Bozic, N., Pujolle, G. & Secci, S. (2016), A tutorial on blockchain and applications to secure network control-planes, *in* 'Smart Cloud Networks & Systems (SCNS)', IEEE, pp. 1–8.

Braden, R., Clark, D. & Shenker, S. (1994), Integrated services in the internet architecture: an overview, Technical report, RFC rpt no.1633.

Budzianowski, P., Casanueva, I., Tseng, B.-H. & Gasic, M. (2018), 'Towards end-to-end multi-domain dialogue modelling'.

Burstein, M. & Pelavin, R. (1984), 'Hierarchical channel router', *Computer-Aided Design* **16**(4), 216–224.

Buszta, K. (2019), Security management, *in* 'Information Security Management', Auerbach Publications, pp. 263–274.

Buterin, V. (2016), Chain interoperability, Technical report, R3 research paper.

Cachin, C. & Vukolić, M. (2017), 'Blockchains consensus protocols in the wild', *arXiv preprint arXiv:1707.01873* .

Cai, W., Wang, Z., Ernst, J. B., Hong, Z., Feng, C. & Leung, V. C. (2018), 'Decentralized applications: The blockchain-empowered software system', *IEEE Access* **6**, 53019–53033.

Calvaresi, D., Marinoni, M., Dragoni, A. F., Hilfiker, R. & Schumacher, M. (2019), 'Real-time multi-agent systems for telerehabilitation scenarios', *Artificial intelligence in medicine* **96**, 217–231.

Cerf, V., Dalal, Y. & Sunshine, C. (1974), Specification of internet transmission control program, Technical report, RFC 675.

Cevallos, A. & Stewart, A. (2020), 'Validator election in nominated proof-of-stake'.

Chakravorty, A. & Rong, C. (2017), Ushare: user controlled social media based on blockchain, *in* 'Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication', ACM, p. 99.

Chauhan, A., Malviya, O. P., Verma, M. & Mor, T. S. (2018), Blockchain and scalability, *in* '2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)', IEEE, pp. 122–128.

CHEN, Z.-d., Zhuo, Y., DUAN, Z.-b. & Kai, H. (2017), 'Inter-blockchain communication', *DEStech Transactions on Computer Science and Engineering* (cst).

Choi, S.-K., Kwon, O.-W. & Kim, Y.-K. (2017), 'Computer-assisted english learning system based on free conversation by topic', *CALL in a climate of change: adapting to turbulent global conditions* p. 79.

Choo, K.-K. R., Ozcan, S., Dehghantanha, A. & Parizi, R. M. (2020), 'Blockchain ecosystem—technological and management opportunities and challenges', *IEEE Transactions on Engineering Management* **67**(4), 982–987.

Chow, J. (2016), 'Btc relay', *btc-relay* .

Christidis, K. & Devetsikiotis, M. (2016), 'Blockchains and smart contracts for the internet of things', *Ieee Access* **4**, 2292–2303.

Çolak, M., Kaya, I., Özkan, B., Budak, A. & Karaşan, A. (2020), 'A multi-criteria evaluation model based on hesitant fuzzy sets for blockchain technology in supply chain management', *Journal of Intelligent & Fuzzy Systems* **38**(1), 935–946.

Conoscenti, M., Vetro, A. & De Martin, J. C. (2016), Blockchain for the internet of things: A systematic literature review, *in* 'Computer Systems and Applications (AICCSA), 2016 IEEE/ACS 13th International Conference of', IEEE, pp. 1–6.

Cosares, S., Kalish, K., Maciura, T. & Spieler, A. C. (2021), Blockchain applications in finance, *in* 'The Emerald Handbook of Blockchain for Business', Emerald Publishing Limited.

Croman, K., Decker, C., Eyal, I., Gencer, A. E., Juels, A., Kosba, A., Miller, A., Saxena, P., Shi, E., Sirer, E. G. et al. (2016), On scaling decentralized blockchains, *in* 'International Conference on Financial Cryptography and Data Security', Springer, pp. 106–125. Christ Church, Barbados.

Czyzewski, A., Dalton, J. & Leuski, A. (2020), Agent dialogue: A platform for conversational information seeking experimentation, *in* 'Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval', pp. 2121–2124.

Darking, M., Whitley, E. A. & Dini, P. (2008), 'Governing diversity in the digital ecosystem', *Communications of the ACM* **51**(10), 137–140.

De Angelis, S., Aniello, L., Baldoni, R., Lombardi, F., Margheri, A. & Sassone, V. (2018), 'Pbft vs proof-of-authority: Applying the cap theorem to permissioned blockchain'.

Deng, L., Chen, H., Zeng, J. & Zhang, L.-J. (2018), Research on cross-chain technology based on sidechain and hash-locking, *in* 'International Conference on Edge Computing', Springer, pp. 144–151. Seattle, WA, USA.

Dennis, R. & Owen, G. (2015), Rep on the block: A next generation reputation system based on the blockchain, *in* '2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)', IEEE, pp. 131–138.

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.

Dinh, T. T. A., Wang, J., Chen, G., Liu, R., Ooi, B. C. & Tan, K.-L. (2017), Blockbench: A framework for analyzing private blockchains, *in* 'Proceedings of the 2017 ACM International Conference on Management of Data', pp. 1085–1100.

Dorri, A., Kanhere, S. S. & Jurdak, R. (2016), 'Blockchain in internet of things: challenges and solutions', *arXiv preprint arXiv:1608.05187* .

Dorri, A., Kanhere, S. S. & Jurdak, R. (2017), Towards an optimized blockchain for iot, *in* 'Proceedings of the Second International Conference on Internet-of-Things Design and Implementation', ACM, pp. 173–178.

Dorri, A., Kanhere, S. S., Jurdak, R. & Gauravaram, P. (2017), Blockchain for iot security and privacy: The case study of a smart home, *in* 'Pervasive Computing and Communications Workshops (PerCom Workshops), 2017 IEEE International Conference on', IEEE, pp. 618–623.

Duffield, E. & Hagan, K. (2014), 'Darkcoin: Peertopeer cryptocurrency with anonymous blockchain transactions and an improved proofofwork system', *bitpaper. info* .

Dujak, D. & Sajter, D. (2019), Blockchain applications in supply chain, *in* 'SMART supply network', Springer, pp. 21–46.

Duong, T., Chepurnoy, A., Fan, L. & Zhou, H.-S. (2018), Twinscoin: A cryptocurrency via proof-of-work and proof-of-stake, *in* 'Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts', ACM, pp. 1–13.

Eberhardt, J. & Tai, S. (2017), On or off the blockchain? insights on off-chaining computation and data, *in* 'European Conference on Service-Oriented and Cloud Computing', Springer, pp. 3–15.

Edlund, J. & Hjalmarsson, A. (2005), Applications of distributed dialogue systems: the kth connector, *in* 'COST278 Final Workshop and ITRW on Applied Spoken Language Interaction in Distributed Environments'.

Eykholt, E., Meredith, L. G. & Denman, J. (2017), 'Rchain architecture documentation', *Retrieve. Jan* **19**, 2019.

Farshidi, S., Jansen, S., España, S. & Verkleij, J. (2020), 'Decision support for blockchain platform selection: Three industry case studies', *IEEE Transactions on Engineering Management* **67**(4), 1109–1128.

Flores, F., Graves, M., Hartfield, B. & Winograd, T. (1988), 'Computer systems and the design of organizational interaction', *ACM Transactions on Information Systems (TOIS)* **6**(2), 153–172.

Franklin, S. (2014), 'History, motivations, and core themes', *The Cambridge handbook of artificial intelligence* pp. 15–33.

Fridgen, G., Radszuwill, S., Urbach, N. & Utz, L. (2018), 'Cross-organizational workflow management using blockchain technology-towards applicability, auditability, and automation'. 51st Annual Hawaii International Conference on System Sciences (HICSS-51).

Gaetani, E., Aniello, L., Baldoni, R., Lombardi, F., Margheri, A. & Sassone, V. (2017), 'Blockchain-based database to ensure data integrity in cloud computing environments'. In Proceedings of the First Italian Conference on Cybersecurity (ITASEC17).

Gervais, A., Karame, G. O., Wüst, K., Glykantzis, V., Ritzdorf, H. & Capkun, S. (2016), On the security and performance of proof of work blockchains, *in* 'Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security', ACM, pp. 3–16.

Ghinita, G., Karras, P., Kalnis, P. & Mamoulis, N. (2009), 'A framework for efficient data anonymization under privacy and accuracy constraints', *ACM Transactions on Database Systems (TODS)* **34**(2), 9.

Giordano, M. T. (2021), Blockchain and the gdpr: New challenges for privacy and security, *in* 'Blockchain, Law and Governance', Springer, pp. 275–286.

Goepel, K. D. (2019), 'Comparison of judgment scales of the analytical hierarchy process—a new approach', *International Journal of Information Technology & Decision Making* **18**(02), 445–463.

Gramoli, V. (2017), 'From blockchain consensus back to byzantine consensus', *Future Generation Computer Systems* pp. 1–20.

Greenspan, G. (2015), 'Multichain private blockchain—white paper', *URl: http://www. multichain. com/download/MultiChain-White-Paper. pdf* .

Gribble, S. D., Brewer, E. A., Hellerstein, J. M. & Culler, D. (2000), Scalable, distributed data structures for internet service construction, *in* 'Proceedings of the 4th conference on Symposium on Operating System Design & Implementation-Volume 4', USENIX Association, p. 22.

Guo, S., Wang, F., Zhang, N., Qi, F. & Qiu, X. (2020*a*), 'Master-slave chain based trusted cross-domain authentication mechanism in iot', *Journal of Network and Computer Applications* **172**, 102812.

Guo, S., Wang, F., Zhang, N., Qi, F. & Qiu, X. (2020*b*), 'Master-slave chain based trusted cross-domain authentication mechanism in iot', *Journal of Network and Computer Applications* **172**, 102812.

Guo, Y. & Liang, C. (2016), 'Blockchain application and outlook in the banking industry', *Financial Innovation* **2**(1), 24.

Halevi, S. & Krawczyk, H. (2006), Strengthening digital signatures via randomized hashing, *in* 'Annual International Cryptology Conference', Springer, pp. 41–59. Santa Barbara, California, USA.

Hari, A. & Lakshman, T. (2016), The internet blockchain: A distributed, tamper-resistant transaction framework for the internet, *in* 'Proceedings of the 15th ACM Workshop on Hot Topics in Networks', ACM, pp. 204–210.

He, W., Akhawe, D., Jain, S., Shi, E. & Song, D. (2014), Shadowcrypt: Encrypted web applications for everyone, *in* 'Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security', ACM, pp. 1028–1039.

He, Y., Li, H., Cheng, X., Liu, Y., Yang, C. & Sun, L. (2018), 'A blockchain based truthful incentive mechanism for distributed p2p applications', *IEEE Access* **6**, 27324–27335.

Henderson, P., Sinha, K., Angelard-Gontier, N., Ke, N. R., Fried, G., Lowe, R. & Pineau, J. (2018), Ethical challenges in data-driven dialogue systems, *in* 'Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society', pp. 123–129.

Hepp, T., Sharinghousen, M., Ehret, P., Schoenhals, A. & Gipp, B. (2018), 'On-chain vs. off-chain storage for supply-and blockchain integration', *it-Information Technology* **60**(5-6), 283–291.

Hertling, S. & Paulheim, H. (2020), 'Dbkwik: extracting and integrating knowledge from thousands of wikis', *Knowledge and Information Systems* **62**(6), 2169–2190.

Heuer, F., Jager, T., Kiltz, E. & Schäge, S. (2016), 'On the selective opening security of practical public-key encryption schemes', *IET Information Security* **10**(6), 304–318.

Hileman, G. & Rauchs, M. (2017), 'Global blockchain benchmarking study. cambridge centre for alternative finance, university of cambridge, 122'.

Hirschberg, J. & Manning, C. D. (2015), 'Advances in natural language processing', *Science* **349**(6245), 261–266.

Hu, F., Hao, Q. & Bao, K. (2014), 'A survey on software-defined network and openflow: From concept to implementation', *IEEE Communications Surveys & Tutorials* **16**(4), 2181–2206.

Hu, X., Morrison, D. M. & Cai, Z. (2013), 'Conversation-based intelligent tutoring system', *Design Recommendations for Intelligent Tutoring Systems: Learner Modeling* **1**, 97–110.

Huang, J.-X., Lee, K.-S., Kwon, O.-W. & Kim, Y.-K. (2017), 'A chatbot for a dialogue-based second language learning system', *CALL in a climate of change: adapting to turbulent global conditions–short papers from EUROCALL* pp. 151–156.

Hughes, D. (n.d.), 'Tempo, our ledger and consensus tech', RADIX DLT Ltd. 2018.

Hunt, P., Konar, M., Junqueira, F. P. & Reed, B. (2010), Zookeeper: Wait-free coordination for internet-scale systems., *in* 'USENIX annual technical conference', Vol. 8, Boston, MA, USA.

Hyeon, J., Oh, K.-J., Kim, Y. J., Chung, H., Kang, B. H. & Choi, H.-J. (2016), Constructing an initial knowledge base for medical domain expert system using induct rdr, *in* '2016 International Conference on Big Data and Smart Computing (BigComp)', IEEE, pp. 408–410.

Idelberger, F., Governatori, G., Riveret, R. & Sartor, G. (2016), Evaluation of logic-based smart contracts for blockchain systems, *in* 'International Symposium on Rules and Rule Markup Languages for the Semantic Web', Springer, pp. 167–183. NY, USA.

Ismailisufi, A., Popović, T., Gligorić, N., Radonjic, S. & Šandi, S. (2020), A private blockchain implementation using multichain open source platform, *in* '2020 24th International Conference on Information Technology (IT)', IEEE, pp. 1–4.

Jeon, S., Doh, I. & Chae, K. (2018), Rmbc: Randomized mesh blockchain using dbft consensus algorithm, *in* 'Information Networking (ICOIN), 2018 International Conference on', IEEE, pp. 712–717.

Jiang, J. J. & Conrath, D. W. (1997), 'Semantic similarity based on corpus statistics and lexical taxonomy', *arXiv preprint cmp-lg/9709008* .

Jiang, J., Versteeg, S., Han, J., Hossain, M. A. & Schneider, J.-G. (2020), 'A positional keyword-based approach to inferring fine-grained message formats', *Future Generation Computer Systems* **102**, 369–381.

Joe, M. M., Ramakrishnan, B. & Das, R. (2016), 'Designing a novel two-tier authentication algorithm for web service architecture', *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)* **8**(9), 67–75.

Johnson, S., Robinson, P. & Brainard, J. (2019), 'Sidechains and interoperability', *arXiv preprint arXiv:1903.04077* .

Joshi, N. (2017), 'Blockchain meets industry 4.0-what happened next?'.

Kahraman, C., Cebeci, U. & Ulukan, Z. (2003), 'Multi-criteria supplier selection using fuzzy ahp', *Logistics information management* .

Kairouz, P., Oh, S. & Viswanath, P. (2014), Extremal mechanisms for local differential privacy, *in* 'Advances in neural information processing systems', pp. 2879–2887.

Kambalyal, C. (2010), '3-tier architecture', *Retrieved On* **2**(34), 2010.

Kan, L., Wei, Y., Muhammad, A. H., Siyuan, W., Gao, L. C. & Kai, H. (2018), A multiple blockchains architecture on inter-blockchain communication, *in* '2018 IEEE international conference on software quality, reliability and security companion (QRS-C)', IEEE, pp. 139–145.

Karayazi, F. & Bereketli, I. (2020), Criteria weighting for blockchain software selection using fuzzy ahp, *in* 'International Conference on Intelligent and Fuzzy Systems', Springer, pp. 608–615.

Keary, M. (2012), 'Knowledge sharing in professions: Roles and identity in expert communities', *Online Information Review* .

Khan, S. N., Loukil, F., Ghedira-Guegan, C., Benkhelifa, E. & Bani-Hani, A. (2021), 'Blockchain smart contracts: Applications, challenges, and future trends', *Peer-to-peer Networking and Applications* pp. 1–25.

Kiayias, A., Russell, A., David, B. & Oliynykov, R. (2017), Ouroboros: A provably secure proof-of-stake blockchain protocol, *in* 'Annual International Cryptology Conference', Springer, pp. 357–388. Santa Barbara, CA, USA.

Kim, S., Kwon, Y. & Cho, S. (2018), A survey of scalability solutions on blockchain, *in* '2018 International Conference on Information and Communication Technology Convergence (ICTC)', IEEE, pp. 1204–1207.

King, S. (2013), Primecoin: Cryptocurrency with prime number proof-of-work, Technical report, Smith + Crown research organization.

King, S. & Nadal, S. (2012), 'Ppcoin: Peer-to-peer crypto-currency with proof-of-stake', *self-published paper, August* **19**.

Kong, X., Zhang, J., Wang, H. & Shu, J. (2019), 'Framework of decentralized multi-chain data management for power systems', *CSEE journal of power and energy systems* **6**(2), 458–468.

Kosba, A., Miller, A., Shi, E., Wen, Z. & Papamanthou, C. (2016), Hawk: The blockchain model of cryptography and privacy-preserving smart contracts, *in* '2016 IEEE symposium on security and privacy (SP)', IEEE, pp. 839–858.

Kratz, J. & Strasser, C. (2014), 'Data publication consensus and controversies', *F1000Research* **3**.

Kreps, J., Narkhede, N., Rao, J. et al. (2011), Kafka: A distributed messaging system for log processing, *in* 'Proceedings of the NetDB', Vol. 11, pp. 1–7.

Kruijff-Korbayová, I., Athanasopoulos, G., Beck, A., Cosi, P., Cuayáhuitl, H., Dekens, T., Enescu, V., Hiolle, A., Kiefer, B., Sahli, H. et al. (2011), An event-based conversational system for the nao robot, *in* 'Proceedings of the Paralinguistic Information and its Integration in Spoken Dialogue Systems Workshop', Springer, pp. 125–132.

Kuo, T.-T., Zavaleta Rojas, H. & Ohno-Machado, L. (2019), 'Comparison of blockchain platforms: a systematic review and healthcare examples', *Journal of the American Medical Informatics Association* **26**(5), 462–478.

Kwon, J. & Buchman, E. (2018), 'A network of distributed ledgers', *Cosmos, dated* pp. 1–41.

Kwon, O.-W., Lee, K., Roh, Y.-H., Huang, J.-X., Choi, S.-K., Kim, Y.-K., Jeon, H. B., Oh, Y. R., Lee, Y.-K., Kang, B. O. et al. (2015), Genietutor: A computer-assisted second-language learning

system based on spoken language understanding, *in* 'Natural language dialog systems and intelligent assistants', Springer, pp. 257–262.

Lai, C., Zhang, M., Cao, J. & Zheng, D. (2019), 'Spir: A secure and privacy-preserving incentive scheme for reliable real-time map updates', *IEEE Internet of Things Journal* **7**(1), 416–428.

Laihonen, H. & Mäntylä, S. (2017), 'Principles of performance dialogue in public administration', *International Journal of Public Sector Management* .

Larimer, D. (2013), Transactions as proof-of-stake, Technical report, Bitcoin Forum, Whitepaper.

Larimer, D. (2014), Delegated proof-of-stake (dpos), Technical report, BitShares Blockchain Foundation.

Larimer, D., Scott, N., Zavgorodnev, V., Johnson, B., Calfee, J. & Vandeberg, M. (2016), 'Steem: An incentivized, blockchain-based social media platform', *March. Self-published* .

Larmuseau, A. & Shila, D. M. (2019), 'Private blockchain configurations for improved iot security', *Blockchain for Distributed Systems Security* pp. 253–274.

Latham, G. P. (2012), *Work motivation: History, theory, research, and practice*, Sage.

Leacock, C., Chodorow, M. & Miller, G. A. (1998), 'Using corpus statistics and wordnet relations for sense identification', *Computational Linguistics* **24**(1), 147–165.

Lee, L. (2016), 'New kids on the blockchain: How bitcoin's technology could reinvent the stock market', *Hastings Business Law Journal* **12**(2).

Lerner, S. D. (2015), 'Rsk white paper overview', *RSK Labs. Online verfügbar unter https://docs. rsk. co/RSK_White_Paper-Overview. pdf, zuletzt geprüft am* **26**, 2019.

Li, M., Weng, J., Yang, A., Lu, W., Zhang, Y., Hou, L., Jia-Nan, L., Xiang, Y. & Deng, R. (2018), 'Crowdbc: A blockchain-based decentralized framework for crowdsourcing', *IEEE Transactions on Parallel and Distributed Systems* .

Lian, H., Qin, Z., He, T. & Luo, B. (2017), Knowledge graph construction based on judicial data with social media, *in* '2017 14th Web Information Systems and Applications Conference (WISA)', IEEE, pp. 225–227.

Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K. & Njilla, L. (2017), Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability, *in*

'Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing', IEEE Press, pp. 468–477.

Lin, I.-C. & Liao, T.-C. (2017), 'A survey of blockchain security issues and challenges.', *IJ Network Security* **19**(5), 653–659.

Liu, J., de la Peña, D. M., Ohran, B. J., Christofides, P. D. & Davis, J. F. (2008), 'A two-tier architecture for networked process control', *Chemical Engineering Science* **63**(22), 5394–5409.

Liu, S., Hu, Y., Zhang, X., Li, Y. & Liu, L. (2020), 'Blockchain service provider selection based on an integrated bwm-entropy-topsis method under an intuitionistic fuzzy environment', *IEEE Access* **8**, 104148–104164.

Ljungqvist, A. & Wilhelm Jr, W. J. (2003), 'Ipo pricing in the dot-com bubble', *The Journal of Finance* **58**(2), 723–752.

Ltd, E. P. (2020), 'The best of british'.
URL: *https://www.effingpot.com/*

Lu, Z., Wang, Q., Qu, G. & Liu, Z. (2018), Bars: a blockchain-based anonymous reputation system for trust management in vanets, *in* '2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)', IEEE, pp. 98–103.

Luu, L., Chu, D.-H., Olickel, H., Saxena, P. & Hobor, A. (2016), Making smart contracts smarter, *in* 'Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security', ACM, pp. 254–269.

Ma, Z., Huang, W., Bi, W., Gao, H. & Wang, Z. (2018), 'A master-slave blockchain paradigm and application in digital rights management', *China Communications* **15**(8), 174–188.

Maček, D. & Alagić, D. (2017), 'Comparisons of bitcoin cryptosystem with other common internet transaction systems by ahp technique', *Journal of Information and Organizational Sciences* **41**(1), 69–87.

Madigan, D. J., Baumann, Z. & Fisher, N. S. (2012), 'Pacific bluefin tuna transport fukushima-derived radionuclides from japan to california', *Proceedings of the National Academy of Sciences* **109**(24), 9483–9486.

Mahto, D., Khan, D. A. & Yadav, D. K. (2016), Security analysis of elliptic curve cryptography and rsa, *in* 'Proceedings of the world congress on engineering', Vol. 1, pp. 419–422.

Manuel, P. D. & AlGhamdi, J. (2003), 'A data-centric design for n-tier architecture', *Information Sciences* **150**(3-4), 195–206.

Maple, C. & Jackson, J. (2018), Selecting effective blockchain solutions, *in* 'European Conference on Parallel Processing', Springer, pp. 392–403.

Mayer, H. (2016), 'Ecdsa security in bitcoin and ethereum: a research survey', *CoinFaabrik, June* **28**(126), 50.

McAfee, A., Brynjolfsson, E., Davenport, T. H., Patil, D. & Barton, D. (2012), 'Big data: the management revolution', *Harvard business review* **90**(10), 60–68.

Melis, A., Mirri, S., Prandi, C., Prandini, M., Salomoni, P. & Callegati, F. (2016), A microservice architecture use case for persons with disabilities, *in* 'International Conference on Smart Objects and Technologies for Social Good', Springer, pp. 41–50. Venice, Italy.

Meneghetti, A., Parise, T., Sala, M. & Taufer, D. (2019), 'A survey on efficient parallelization of blockchain-based smart contracts', *arXiv preprint arXiv:1904.00731* .

Merminod, V., Rowe, F. & Teeni, D. (2012), 'Knowledge sharing and maturation in circles of trust: The case of new product development'.

Meshcheryakov, A., Ivanov, S. et al. (2020), 'Ethereum as a hedge: the intraday analysis', *Economics Bulletin* **40**(1), 101–108.

Milutinovic, M., He, W., Wu, H. & Kanwal, M. (2016), Proof of luck: An efficient blockchain consensus protocol, *in* 'Proceedings of the 1st Workshop on System Software for Trusted Execution', ACM, p. 2.

Mingxiao, D., Xiaofeng, M., Zhe, Z., Xiangwei, W. & Qijun, C. (2017), A review on consensus algorithm of blockchain, *in* '2017 IEEE international conference on systems, man, and cybernetics (SMC)', IEEE, pp. 2567–2572.

Mmbaga, N. A. (2018), The roles of reward system in retaining competent employees at Wiafrica Tanzania Limited, PhD thesis.

Mohanta, B. K., Jena, D., Panda, S. S. & Sobhanayak, S. (2019), 'Blockchain technology: A survey on applications and security privacy challenges', *Internet of Things* **8**, 100107.

Mourao, M., Cassaca, R. & Mamede, N. (2004), An independent domain dialogue system through a service manager, *in* 'International Conference on Natural Language Processing (in Spain)', Springer, pp. 161–171.

Münsing, E., Mather, J. & Moura, S. (2017), Blockchains for decentralized optimization of energy resources in microgrid networks, *in* 'Control Technology and Applications (CCTA), 2017 IEEE Conference on', IEEE, pp. 2164–2171.

Nakamoto, S. (n.d.), 'Bitcoin: A peer-to-peer electronic cash system', `http://bitcoin.org/bitcoin.pdf`. 2008.

Nakano, M., Hasegawa, Y., Funakoshi, K., Takeuchi, J., Torii, T., Nakadai, K., Kanda, N., Komatani, K., Okuno, H. G. & Tsujino, H. (2011), 'A multi-expert model for dialogue and behavior control of conversational robots and agents', *Knowledge-Based Systems* **24**(2), 248–256.

Nakano, M. & Komatani, K. (2020), 'A framework for building closed-domain chat dialogue systems', *Knowledge-Based Systems* **204**, 106212.

Namasudra, S., Deka, G. C., Johri, P., Hosseinpour, M. & Gandomi, A. H. (2021), 'The revolution of blockchain: State-of-the-art and research challenges', *Archives of Computational Methods in Engineering* **28**(3), 1497–1515.

Neter, J., Kutner, M. H., Nachtsheim, C. J. & Wasserman, W. (1996), 'Applied linear statistical models'.

Nguyen, A. & Wobcke, W. (2005), An agent-based approach to dialogue management in personal assistants, *in* 'Proceedings of the 10th international conference on Intelligent user interfaces', pp. 137–144.

Nguyen, G.-T. & Kim, K. (2018), 'A survey about consensus algorithms used in blockchain', *Journal of Information processing systems* **14**(1), 101–128.

Nguyen, Q. K. (2016), Blockchain-a financial technology for future sustainable development, *in* 'Green Technology and Sustainable Development (GTSD), International Conference on', IEEE, pp. 51–54.

Nian, F., Liu, R. & Cong, A. (2020), 'An incentive mechanism model based on the correlation between neighbor behavior and distance', *International Journal of Modern Physics C* **31**(11), 2050161.

Nick, J., Poelstra, A. & Sanders, G. (2020), 'Liquid: A bitcoin sidechain', *Liquid white paper. URL https://blockstream. com/assets/downloads/pdf/liquid-whitepaper. pdf* .

Ongaro, D. & Ousterhout, J. (2015), 'The raft consensus algorithm'.

Pal, M. (2017), 'Alphablock', *Available at SSRN 3070978* .

Pan, J. Z., Pavlova, S., Li, C., Li, N., Li, Y. & Liu, J. (2018), Content based fake news detection using knowledge graphs, *in* 'International semantic web conference', Springer, pp. 669–683.

Papangelis, A., Wang, Y.-C., Molino, P. & Tur, G. (2019), 'Collaborative multi-agent dialogue model training via reinforcement learning', *arXiv preprint arXiv:1907.05507* .

Pass, R. & Shi, E. (2017), Hybrid consensus: Efficient consensus in the permissionless model, *in* 'LIPIcs-Leibniz International Proceedings in Informatics', Vol. 91, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Pei, J., Ren, P. & de Rijke, M. (2019), 'A modular task-oriented dialogue system using a neural mixture-of-experts', *arXiv preprint arXiv:1907.05346* .

Peikert, C. (2014), Lattice cryptography for the internet, *in* 'International Workshop on Post-Quantum Cryptography', Springer, Cham, pp. 197–219.

Peng, K., Li, M., Huang, H., Wang, C., Wan, S. & Choo, K.-K. R. (2021), 'Security challenges and opportunities for smart contracts in internet of things: A survey', *IEEE Internet of Things Journal* **8**(15), 12004–12020.

Perera, R. & Nand, P. (2017), 'Recent advances in natural language generation: A survey and classification of the empirical literature', *Computing and Informatics* **36**(1), 1–32.

Peters, G., Panayi, E. & Chapelle, A. (2015), 'Trends in cryptocurrencies and blockchain technologies: a monetary theory and regulation perspective', *Journal of Financial Perspectives* **3**(3).

Pinyaphat, T. (2018), Blockchain: Challenges and applications, *in* '2018 International Conference on Information Networking (ICOIN)', IEEE, pp. 473–475.

Poon, J. & Dryja, T. (2016), 'The bitcoin lightning network: Scalable off-chain instant payments', *See https://lightning. network/lightning-network-paper. pdf* .

Pop, C., Cioara, T., Antal, M., Anghel, I., Salomie, I. & Bertoncini, M. (2018), 'Blockchain based decentralized management of demand response programs in smart energy grids', *Sensors* **18**(1), 162.

Popov, S. (n.d.*a*), 'Byteball wiki', `https://byteball.org/`. 2017.

Popov, S. (n.d.*b*), 'The tangle', `https://iotchain.io/`. 2018.

Pradhan, R., Bykau, S. & Prabhakar, S. (2017), Staging user feedback toward rapid conflict resolution in data fusion, *in* 'Proceedings of the 2017 ACM International Conference on Management of Data', pp. 603–618.

Qin, J., Ma, Q., Shi, Y. & Wang, L. (2016), 'Recent advances in consensus of multi-agent systems: A brief survey', *IEEE Transactions on Industrial Electronics* **64**(6), 4972–4983.

Rahmadika, S., Ramdania, D. R. & Harika, M. (2018), 'Security analysis on the decentralized energy trading system using blockchain technology', *Jurnal Online Informatika* **3**(1), 44–47.

Rathee, G., Ahmad, F., Kurugollu, F., Azad, M. A., Iqbal, R. & Imran, M. (2020), 'Crt-biov: a cognitive radio technique for blockchain-enabled internet of vehicles', *IEEE Transactions on Intelligent Transportation Systems* .

Raval, S. (2016), *Decentralized Applications: Harnessing Bitcoin's Blockchain Technology*, " O'Reilly Media, Inc.".

Reed, J. (2017), 'Litecoin: An introduction to litecoin cryptocurrency and litecoin mining'.

Reza M. Parizi, e. a. (2018), Smart contract programming languages on blockchains: An empirical evaluation of usability and security, *in* 'International Conference on Blockchain', Springer, pp. 75–91.

Ruta, M., Scioscia, F., Gramegna, F., Ieva, S., Di Sciascio, E. & De Vera, R. P. (2018), 'A knowledge fusion approach for context awareness in vehicular networks', *IEEE Internet of Things Journal* **5**(4), 2407–2419.

Saaty, T. L. (2008), 'Decision making with the analytic hierarchy process', *International journal of services sciences* **1**(1), 83–98.

Sáez, M. (2020), 'Blockchain-enabled platforms: Challenges and recommendations.', *International Journal of Interactive Multimedia & Artificial Intelligence* **6**(3).

Safa, N. S., Von Solms, R. & Furnell, S. (2016), 'Information security policy compliance model in organizations', *computers & security* **56**, 70–82.

Sankar, L. S., Sindhu, M. & Sethumadhavan, M. (2017), Survey of consensus protocols on blockchain applications, *in* 'Advanced Computing and Communication Systems (ICACCS), 2017 4th International Conference on', IEEE, pp. 1–5.

Shahin, N., Ali, R., Nam, S. Y. & Kim, Y.-T. (2018), Performance evaluation of centralized and distributed control methods for efficient registration of massive iot devices, *in* '2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN)', IEEE, pp. 314–319.

Sharma, P., Jindal, R. & Borah, M. D. (2021), 'A review of blockchain-based applications and challenges', *Wireless Personal Communications* pp. 1–43.

Shibata, M., Nishiguchi, T. & Tomiura, Y. (2009), 'Dialog system for open-ended conversation using web documents', *Informatica* **33**(3).

Shih, H.-S., Shyur, H.-J. & Lee, E. S. (2007), 'An extension of topsis for group decision making', *Mathematical and computer modelling* **45**(7-8), 801–813.

Shuchun, L., Mengyang, L., Shixian, W. & Zhenqin, X. (2000), 'The design and implementation of the browser/server mode mis [j]', *Computer Engineering and Applications* **6**, 038.

Siewert, S. (2018), Why software engineers and developers should care about blockchain technology, Technical report, Applications beyond digital currency.

Singh, A., Click, K., Parizi, R. M., Zhang, Q., Dehghantanha, A. & Choo, K.-K. R. (2020), 'Sidechain technologies in blockchain networks: An examination and state-of-the-art review', *Journal of Network and Computer Applications* **149**, 102471.

Smaragdakis, G., Laoutaris, N., Oikonomou, K., Stavrakakis, I. & Bestavros, A. (2014), 'Distributed server migration for scalable internet service deployment', *IEEE/ACM Transactions on Networking (TON)* **22**(3), 917–930.

Smirnov, A. & Levashova, T. (2019), 'Knowledge fusion patterns: A survey', *Information Fusion* **52**, 31–40.

Song, J. C., Demir, M. A., Prevost, J. J. & Rad, P. (2018), Blockchain design for trusted decentralized iot networks, *in* '2018 13th Annual Conference on System of Systems Engineering (SoSE)', IEEE, pp. 169–174.

Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J.-Y., Gao, J. & Dolan, B. (2015), 'A neural network approach to context-sensitive generation of conversational responses', *arXiv preprint arXiv:1506.06714* .

Stewart, A. & Kokoris-Kogia, E. (2020), 'Grandpa: a byzantine finality gadget', *arXiv preprint arXiv:2007.01560* .

Su, H., Li, F., Liang, Q., Miao, C. X. & Xin, L. (2018), 'Trustable web searching verification in a blockchain'. US Patent App. 15/349,299.

Su, Y. & Huang, J. (2012), 'Two consensus problems for discrete-time multi-agent systems with switching network topology', *Automatica* **48**(9), 1988–1997.

Sukhwani, H., Martínez, J. M., Chang, X., Trivedi, K. S. & Rindos, A. (2017), Performance modeling of pbft consensus process for permissioned blockchain network (hyperledger fabric), *in* '2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)', IEEE, pp. 253–255.

Swan, M. (2015), *Blockchain: Blueprint for a new economy*, " O'Reilly Media, Inc.".

Syed, T. A., Alzahrani, A., Jan, S., Siddiqui, M. S., Nadeem, A. & Alghamdi, T. (2019), 'A comparative analysis of blockchain architecture and its applications: Problems and recommendations', *IEEE access* **7**, 176838–176869.

Tamjidyamcholo, A., Baba, M. S. B., Shuib, N. L. M. & Rohani, V. A. (2014), 'Evaluation model for knowledge sharing in information security professional virtual community', *Computers & Security* **43**, 19–34.

Tang, J., Zhao, T., Xiong, C., Liang, X., Xing, E. P. & Hu, Z. (2019), 'Target-guided open-domain conversation', *arXiv preprint arXiv:1905.11553* .

Tang, W. & Daoutidis, P. (2017), 'Distributed/hierarchical control architecture design', *IFAC-PapersOnLine* **50**(1), 12015–12020.

Terzis, A., Wang, L., Ogawa, J. & Zhang, L. (1999), 'A two-tier resource management model for the internet', *GLOBECOM-NEW YORK-* **3**, 1779–1791.

Thomas, S. & Schwartz, E. (2015), 'A protocol for interledger payments', *URL https://interledger. org/interledger. pdf* .

Traum, D. R. & Larsson, S. (2003), The information state approach to dialogue management, *in* 'Current and new directions in discourse and dialogue', Springer, pp. 325–353.

Twinword (2021).

URL: *https://www.twinword.com/api/text-similarity.php*

Upadhyaya, S., Rao, H. R. & Padmanabhan, G. (2011), Secure knowledge management, *in* 'Encyclopedia of Knowledge Management, Second Edition', IGI Global, pp. 1429–1437.

ur Rehman, M. H., Salah, K., Damiani, E. & Svetinovic, D. (2020), Towards blockchain-based reputation-aware federated learning, *in* 'IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)', IEEE, pp. 183–188.

Urgaonkar, B., Pacifici, G., Shenoy, P., Spreitzer, M. & Tantawi, A. (2005), An analytical model for multi-tier internet services and its applications, *in* 'ACM SIGMETRICS Performance Evaluation Review', Vol. 33, ACM, pp. 291–302.

Vacca, A., Di Sorbo, A., Visaggio, C. A. & Canfora, G. (2021), 'A systematic literature review of blockchain and smart contract development: Techniques, tools, and open challenges', *Journal of Systems and Software* **174**, 110891.

Van Schewick, B. (2012), *Internet architecture and innovation*, Mit Press.

Velliangiri, S. (2020), Blockchain technology: Challenges and security issues in consensus algorithm, *in* '2020 International Conference on Computer Communication and Informatics (ICCCI)', IEEE, pp. 1–8.

Venkatesh, A., Khatri, C., Ram, A., Guo, F., Gabriel, R., Nagar, A., Prasad, R., Cheng, M., Hedayatnia, B., Metallinou, A. et al. (2018), 'On evaluating and comparing open domain dialog systems', *arXiv preprint arXiv:1801.03625* .

Viriyasitavat, W., Da Xu, L., Bi, Z. & Sapsomboon, A. (2018), 'Blockchain-based business process management (bpm) framework for service composition in industry 4.0', *Journal of Intelligent Manufacturing* pp. 1–12.

Viriyasitavat, W. & Hoonsopon, D. (2018), 'Blockchain characteristics and consensus in modern business processes', *Journal of Industrial Information Integration* .

Voigt, P. & Von dem Bussche, A. (2017), 'The eu general data protection regulation (gdpr)', *A Practical Guide, 1st Ed., Cham: Springer International Publishing* **10**(3152676), 10–5555.

Vokerla, R. R., Shanmugam, B., Azam, S., Karim, A., De Boer, F., Jonkman, M. & Faisal, F. (2019), An overview of blockchain applications and attacks, *in* '2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)', IEEE, pp. 1–6.

Vukolić, M. (2015), The quest for scalable blockchain fabric: Proof-of-work vs. bft replication, *in* 'International Workshop on Open Problems in Network Security', Springer, pp. 112–125.

Walport, M. (2016), 'Distributed ledger technology: Beyond blockchain', *UK Government Office for Science* .

Wang, D., Zhou, J., Wang, A. & Finestone, M. (2018), 'Loopring: A decentralized token exchange protocol', *URL https://github. com/Loopring/whitepaper/blob/master/en_whitepaper. pdf* .

Wang, M., Duan, M. & Zhu, J. (2018), Research on the security criteria of hash functions in the blockchain, *in* 'Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts', pp. 47–55.

Wen, T.-H., Vandyke, D., Mrksic, N., Gasic, M., Rojas-Barahona, L. M., Su, P.-H., Ultes, S. & Young, S. (2016), 'A network-based end-to-end trainable task-oriented dialogue system', *arXiv preprint arXiv:1604.04562* .

Wood, G. (2016), 'Polkadot: Vision for a heterogeneous multi-chain framework', *White Paper* **21**, 2327–4662.

Wurster, S., Böhmecke-Schwafert, M., Hofmann, F. & Blind, K. (2018), Born global market dominators and implications for the blockchain avantgarde, *in* 'Corporate and Global Standardization Initiatives in Contemporary Society', IGI Global, pp. 86–115.

Xia, Q., Sifah, E. B., Asamoah, K. O., Gao, J., Du, X. & Guizani, M. (2017), 'Medshare: Trust-less medical data sharing among cloud service providers via blockchain', *IEEE Access* **5**, 14757–14767.

Xiao, X., Wang, G. & Gehrke, J. (2011), 'Differential privacy via wavelet transforms', *IEEE Transactions on knowledge and data engineering* **23**(8), 1200–1214.

Xie Zhuopeng, Ding Ying, e. a. (n.d.), 'Iot chain: A high-security lite iot os', `https://iotchain.io/`. 2015.

Xu, X., Weber, I. & Staples, M. (2019), *Architecture for blockchain applications*, Springer.

Yan, Y. Y. & Zhi, T. (2005), 'A survey of the research on digital rights management [j]', *Chinese Journal of Computers* **28**(12), 1957–1968.

Yang, W., Aghasian, E., Garg, S., Herbert, D., Disiuta, L. & Kang, B. (2019), 'A survey on blockchain-based internet service architecture: requirements, challenges, trends, and future', *IEEE Access* **7**, 75845–75872.

Yang, W., Garg, S., Raza, A., Herbert, D. & Kang, B. (2018), Blockchain: Trends and future, *in* 'Pacific Rim Knowledge Acquisition Workshop', Springer, pp. 201–210. Nanjing, China.

Yli-Huumo, J., Ko, D., Choi, S., Park, S. & Smolander, K. (2016), 'Where is current research on blockchain technology?—a systematic review', *PloS one* **11**(10), e0163477.

Young, I. (2018), 'Dogecoin: A brief overview & survey', *Available at SSRN 3306060* .

Zhang, Y., Deng, R., Liu, X. & Zheng, D. (2018), 'Outsourcing service fair payment based on blockchain and its applications in cloud computing', *IEEE Transactions on Services Computing* .

Zhang, Z., Wang, F., Zhong, C. & Ma, H. (n.d.), Grid terminal data security management mechanism based on master-slave blockchain, *in* '2020 5th International Conference on Computer and Communication Systems (ICCCS)', IEEE, pp. 67–70.

Zhang, Z., Wu, Y., Zhao, H., Li, Z., Zhang, S., Zhou, X. & Zhou, X. (2020), Semantics-aware bert for language understanding, *in* 'Proceedings of the AAAI Conference on Artificial Intelligence', Vol. 34, pp. 9628–9635.

Zhao, J. L., Fan, S. & Yan, J. (2016), 'Overview of business innovations and research opportunities in blockchain and introduction to the special issue'.

Zhao, Z., Han, S.-K. & So, I.-M. (2018), 'Architecture of knowledge graph construction techniques', *International Journal of Pure and Applied Mathematics* **118**(19), 1869–1883.

Zhaoyang, D., Fengji, L. & Liang, G. (2018), 'Blockchain: a secure, decentralized, trusted cyber infrastructure solution for future energy systems', *Journal of Modern Power Systems and Clean Energy* **6**(5), 958–967.

Zheng, Y., Ma, J. & Wang, L. (2017), 'Consensus of hybrid multi-agent systems', *IEEE transactions on neural networks and learning systems* **29**(4), 1359–1365.

Zheng, Z., Xie, S., Dai, H., Chen, X. & Wang, H. (2017), An overview of blockchain technology: Architecture, consensus, and future trends, *in* '2017 IEEE international congress on big data (BigData congress)', IEEE, pp. 557–564.

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

Zheng, Z., Xie, S., Dai, H.-N. & Wang, H. (2016), 'Blockchain challenges and opportunities: A survey', *International Journal of Web and Grid Services* **1**, 1–25.

Zhou, B., Pei, J. & Luk, W. (2008), 'A brief survey on anonymization techniques for privacy preserving publishing of social network data', *ACM Sigkdd Explorations Newsletter* **10**(2), 12–22.

Zhou, K., Zhao, W. X., Bian, S., Zhou, Y., Wen, J.-R. & Yu, J. (2020), Improving conversational recommender systems via knowledge graph based semantic fusion, *in* 'Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining', pp. 1006–1014.

Zhou, X., Li, P., Zeng, Y., Fan, X., Liu, P. & Miyazaki, T. (2021), 'A fast algorithm for liquid voting on blockchain', *IEICE TRANSACTIONS on Information and Systems* **104**(8), 1163–1171.

Zmaznev, E. et al. (2018), Bitcoin and ethereum evolution, Technical report, CENTRIA UNIVERSITY OF APPLIED SCIENCES, Thesis.

Zyskind, G., Nathan, O. et al. (2015), Decentralizing privacy: Using blockchain to protect personal data, *in* '2015 IEEE Security and Privacy Workshops', IEEE, pp. 180–184.