



UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

**Using distributed ledger technology for digital forensic
investigation purposes on tendering projects**

by

Pardon Takalani Ramazhamba

(19193867)

Submitted in partial fulfillment of the requirements for the degree

Master of Science (Computer Science)

in the

Department of Computer Science

Faculty of Engineering, Built Environment and Information Technology

University of Pretoria

Republic of South Africa

2022

Summary

USING DISTIRBUTED LEDGER TECHNOLOGY FOR DIGITAL FORENSIC INVESTIGATION PURPOSES ON TENDERING PROJECTS

by

Pardon Takalani Ramazhamba

(19193867)

Supervisor: Prof. H.S. Venter
Department: Computer Science
University: University of Pretoria
Degree: Master of Science (Computer Science)
Keywords: Tendering system, Blockchain technology, information security, project information, sharing tendering project

The tendering system used by the South African Government is regarded as a central method used by the organs of state to procure goods and services, including delivering some services to citizens with the aim of promoting social industrial, or environmental policies. Some of these projects are distributed using a tendering system aimed at developing and empowering the surrounding communities. Hence, the tendering system used by these organs of state should be fair, transparent, competitive, cost-effective, equitable, and free from corruption. However, the mismanagement of the tendering system might lead to interruption of operations, poor product quality, late service delivery, rising costs, and most importantly, fraud and corruption. The use of paperwork to share project information might also lead to the mismanagement of the tendering project because it might contribute towards illicit altering of project information during the process. This might also affect the fairness, transparency, data integrity, and competitiveness of the tendering system used by the South African Local Government. Additionally, the process of investigating any fraudulent activity is nearly impossible with the current paper-based tendering system.

The purpose of this study is to implement a Blockchain prototype that can be used to securely share project information with all the participants that have an interest in the tendering project. This Blockchain prototype is called the Share Tendering Project (ShareTendPro) network. Diagrams were used to visualise the design of the ShareTendPro network.

It is recommended that the use of the ShareTendPro network will enable various participants to have access to project information in real-time, allowing them to have access to the entire project history regardless of their geographical location. Access to real-time data would imply that the ShareTendPro network will also promote real-time auditing and digital forensic investigations because both auditors and investigators will have access to the project information of their interest in real-time. Additionally, the project information stored within the ShareTendPro network can also be regarded as credible digital evidence because it is immutable by default. Furthermore, the ShareTendPro network also seeks to reduce human interactions (i.e., human errors) by automating some of the processes within the network while assuring that all data is stored in a digital forensically sound format.

Table of contents

Summary	ii
Table of contents	iv
Lists of figures	xi
List of tables	xv
List of abbreviations and meaning	xvi
Declaration	xvii
Dedication	xviii
Acknowledgment	xix
1. Chapter 1: Introduction	1
1.1 Introduction	1
1.2 Problem statement	2
1.3 Significance of the study	3
1.4 Research objectives	4
1.5 Research Methodology	4
1.6 Scope and motivation of the study	6
1.7 Layout of the study	9
1.8 Conclusion.....	11
2. Chapter 2: Tendering system landscape in the South African Local Government	12
2.1 Introduction	12
2.2 Overview of the supply chain management used by SALG	12
2.2.1 Conventional	15
2.2.1.1 Open tender	15
2.2.1.2 Selective tender	16
2.2.1.3 Negotiated tender.....	16
2.2.2 Non-conventional.....	17
2.2.2.1 Integrated.....	17
2.2.2.2 Management-oriented.....	18
2.2.2.3 Collaborative	18
2.2.3 Stages in the tendering process.....	19
2.2.3.1 Request for invitation of tenders.....	19
2.2.3.2 Calling for tenders.....	20
2.2.3.3 Submission and receipt of tenders	20
2.2.3.4 Open of tenders	21
2.2.3.5 Assessing of tenders.....	21

2.2.3.6	Awarding of tenders.....	22
2.2.4	Legislative framework.....	22
2.2.4.1	Constitution.....	23
2.2.4.2	Municipal Finance Management Act (MFMA), 2003.....	23
2.2.4.3	Preferential Procurement Policy Framework Act (PPPFA), 2000.....	23
2.2.5	Pillars of the procurement process.....	24
2.2.5.1	Value for money.....	24
2.2.5.2	Open and effective competition.....	25
2.2.5.3	Ethics and fair dealings.....	25
2.2.5.4	Accounting and reporting.....	25
2.2.5.5	Equity.....	25
2.2.6	Role players of the Supply Chain Management.....	26
2.2.6.1	The Municipal Council.....	26
2.2.6.2	The Accounting Officer.....	26
2.2.6.3	Municipal SCM Unit.....	26
2.3	Importance of monitoring tender projects.....	27
2.4	Conclusion.....	28
3.	Chapter 3: Distributed ledger technology and security landscape	29
3.1	Introduction.....	29
3.2	Information security.....	30
3.2.1	Confidentiality.....	30
3.2.2	Integrity.....	31
3.2.3	Availability.....	31
3.2.4	Identification and authentication.....	31
3.2.5	Authorisation.....	32
3.2.6	Non-repudiation.....	32
3.3	Overview of ledger system.....	32
3.3.1	Centralised ledger system.....	33
3.3.2	Distributed ledger system.....	34
3.3.3	Consensus algorithms overview.....	36
3.3.3.1	Mining consensus algorithms.....	37
3.3.3.2	Non-mining consensus algorithms.....	42
3.3.4	Distributed ledger technology: the Blockchain data structure.....	44
3.3.5	Blockchain frameworks for developing distributed applications.....	52
3.3.5.1	Bitcoin framework.....	53
3.3.5.2	Ripple framework.....	53

3.3.5.3	Ethereum framework.....	53
3.3.5.4	Hyperledger framework.....	54
3.3.5.5	Selecting a particular Blockchain framework.....	54
3.4	Related work	56
3.5	Conclusion.....	59
4.	Chapter 4: A model for securing and distributing tendering project information	60
4.1	Introduction	60
4.2	The current project information-sharing concept.....	60
4.3	Scenarios	62
4.3.1	Scenario 1.....	62
4.3.2	Scenario 2.....	63
4.3.3	Scenario 3.....	63
4.4	Proposed model.....	63
4.4.1	Identifying actors	65
4.4.1.1	Main actors	66
4.4.1.1.1	District Municipalities	66
4.4.1.1.2	Local Municipalities	66
4.4.1.1.3	Communities.....	67
4.4.1.1.4	Suppliers.....	67
4.4.1.2	Additional actors.....	67
4.4.1.2.1	Auditors.....	68
4.4.1.2.2	Investigators.....	68
4.4.2	Establishing the gateway	69
4.4.3	Establishing the ShareTendPro Blockchain network	71
4.4.4	The ShareTendPro model as an integrated whole.....	72
4.5	Conclusion.....	75
5.	Chapter 5: ShareTendPro model design	76
5.1	Introduction	76
5.2	Requirements Specifications.....	76
5.2.1	Functional requirements.....	76
5.2.2	Non-functional requirements	77
5.3	Model design.....	78
5.3.1	Architectural design (Model architecture).....	79
5.3.2	Behavioural design.....	81
5.3.2.1	Use-case diagram	82
5.3.2.2	State diagram	83

5.3.2.3	Information flow	85
5.3.2.3.1	Overview of the information flow	85
5.3.2.3.2	Detailed information flow for transaction 1	89
5.3.2.3.3	Detailed information flow for transaction 2	90
5.3.2.3.4	Detailed information flow for transaction 3	91
5.3.2.3.5	Detailed information flow for transaction 4	93
5.3.2.3.6	Detailed information flow for transaction 5	94
5.3.2.3.7	Detailed information flow for transaction 6	95
5.3.2.3.8	Overall detailed information flow	96
5.3.3	Structural design	98
5.3.3.1	Class diagram	98
5.3.3.2	Package diagram	100
5.4	Conclusion	101
6.	Chapter 6: Implementation of the ShareTendPro prototype	102
6.1	Introduction	102
6.2	Tools used to develop the prototype	102
6.3	ShareTendPro model implementation	103
6.3.1	ShareTendPro network topology	105
6.3.2	Chaincode development	107
6.4	Summary	108
7.	Chapter 7: ShareTendPro prototype results	109
7.1	Introduction	109
7.2	Scenario	109
7.3	ShareTendPro network results	115
7.3.1	Establish Blockchain network	116
7.3.2	Interacting with Blockchain network	117
7.3.2.1	Create tender	118
7.3.2.2	Query tender	119
7.3.2.3	Update tender	121
7.3.2.4	Delete tender	123
7.3.2.5	Query tender history	123
7.4	Conclusion	124
8.	Chapter 8: Critical Evaluation	126
8.1	Introduction	126
8.2	Benefits of the research	126
8.2.1	Primary benefits	127

8.2.1.1	Distributed nature of the ShareTendPro network.....	127
8.2.1.2	Enhanced information security within the ShareTendPro prototype	128
8.2.1.3	Greater transparency over project information	129
8.2.1.4	Automated transactions	129
8.2.2	Secondary benefits.....	130
8.2.2.1	Time-efficient.....	130
8.2.2.2	Reduction of cost	131
8.2.2.3	Credible evidence.....	132
8.2.2.4	Promotes real-time auditing and investigations.....	132
8.2.2.5	Enforces trust.....	133
8.2.2.6	Improved collaboration.....	133
8.3	Shortcomings of the research.....	134
8.3.1	Storage constraints	134
8.3.2	Off-chain data storage	135
8.3.3	Scalability	135
8.3.4	Lack of political will	135
8.4	Conclusion.....	136
9.	Chapter 9: Conclusion.....	137
9.1	Introduction	137
9.2	Summary of all the chapters	137
9.3	Recap of the problem statement.....	138
9.4	Future work.....	139
9.5	Conclusion.....	140
	References.....	142
A.	Appendix A: Directed Acyclic Graph	155
A.1	IOTA.....	155
A.2	Hashgraph	157
B.	Appendix B: Detailed implementation of the ShareTendPro prototype	158
B.1	Introduction	158
B.2	Tools used to develop the prototype.....	158
B.2.1	Hardware requirements.....	158
B.2.2	Software requirements	159
B.3	ShareTendPro model implementation	164
B.3.1	ShareTendPro network topology	166
B.3.1.1	Crypto-config	170
B.3.1.1.1	Overview of the ShareTendPro network	171

B.3.1.1.2	Crypto-config.yaml file.....	172
B.3.1.1.3	Crypto-config folder.....	177
B.3.1.2	Configtx.....	181
B.3.1.2.1	Channel members.....	182
B.3.1.2.2	Raft ordering service.....	185
B.3.1.2.3	Channel profiles.....	187
B.3.1.2.4	Channel-artifacts folder.....	189
B.3.1.3	Docker-compose.....	190
B.3.1.3.1	Certificate Authority services.....	191
B.3.1.3.2	Ordering node services.....	193
B.3.1.3.3	CouchDB services.....	195
B.3.1.3.4	Peer node services.....	197
B.3.2	Chaincode development.....	200
B.3.2.1	Tender structure.....	201
B.3.2.2	Invoke functions.....	202
B.3.2.2.1	InitLedger.....	203
B.3.2.2.2	CreateTender.....	204
B.3.2.2.3	QueryTender.....	205
B.4	Summary.....	205
C.	Appendix C: ShareTendPro network results.....	206
C.1	Establish Blockchain network.....	206
C.2	Create communication channel.....	207
C.2.1	Create channel.....	208
C.2.2	Join channel.....	209
C.2.3	Update anchor peers.....	210
C.3	Deploy smart-contract.....	211
C.3.1	Install chaincode.....	212
C.3.2	Approve chaincode.....	213
C.3.3	Commit chaincode.....	215
C.4	Update tender.....	216
C.4.1	Update Tender Supplier.....	217
C.4.2	Update Tender Report.....	217
C.5	Delete tender.....	218
D.	Derived Publications.....	220
D.1.	A distributed model for sharing tendering problem information in the South African Local Government. In the CSIR Emerging Researchers Symposium (CSIR-ERS) Poster.....	221

D.2. A distributed model for sharing tendering problem information in the South African Local Government. In the 21st International Conference on Cyberworlds (CW2022)	226
1. Identify the actors of the proposed model	233
2. Establishing the gateway that will be used to identify and authorise actors as they interact with project information.....	233
3. Establishing the Blockchain network that can be used by the proposed model to securely store and share project information.....	233
4. Defining the ShareTendPro model, which is the integration of steps 1–3 above.	233
12. Step 12: Peter submitted a falsified progress report of project X to Martha (DM_NO) at the DM.	239
13. Step 13: The tendering committee of the DM assesses all the suppliers who applied for project Y and submits the results of the assessment to the DM.....	239
14. Step 14: the DM awards tendering project Y to Supplier S based on the outcome of the assessment which was motivated by the information provided by the supplier and confirmed by Peter who works at the LM.....	239
D.3. Using distributed ledger technology for digital forensic investigation purposes on tendering projects. International Journal of Information Technology (IJIT)	248
D.4. ShareCrime: A Blockchain concept for sharing criminal information in Criminal Justice Systems (submitted to the IFIP WG 11.9 International Conference on Digital Forensics)	289
A. Conceptualising the South African criminal justice process	292
B. Adopted technology description	295
A. Identifying users/agents	298
B. Establishing the application mechanisms	298
C. Establish the services mechanisms.....	299
D. Blockchain network.....	300
E. ShareCrimE high-level model.....	301
A. ShareCrimE model information flow	302
B. Sequence diagram of the ShareCrimE Application channel	303
C. Sequence diagram of the ShareCrimE main channel	304
A. Benefits of the research.....	305
B. Shortcomings of the research.....	307

Lists of figures

FIGURE 1.1 RESEARCH METHODOLOGY.....	5
FIGURE 1.2 COMMUNICATION CONCEPT.....	7
FIGURE 2.1 LOCAL GOVERNMENT PROJECTS CONCEPT.....	13
FIGURE 2.2 SCM PROCESSES AND PROCUREMENT CONCEPT.....	14
FIGURE 2.3 STAGES OF THE TENDERING PROCESS.....	19
FIGURE 2.4 LOCAL GOVERNMENT PROCUREMENT LEGISLATION.....	23
FIGURE 2.5 PILLARS OF THE PROCUREMENT PROCESS.....	24
FIGURE 2.6 SCM ROLE PLAYERS.....	26
FIGURE 3.1 CENTRALIZED LEDGER SYSTEM.....	34
FIGURE 3.2 DISTRIBUTED LEDGER SYSTEM.....	35
FIGURE 3.3 LEDGER COMPONENTS [62].....	46
FIGURE 3.4 INFORMATION STORED IN WORLD STATE AND BLOCKCHAIN.....	47
FIGURE 3.5 TRANSITION OF “WORLD STATE” AND BLOCKCHAIN.....	48
FIGURE 3.6 BLOCKCHAIN DATA STRUCTURE.....	49
FIGURE 3.7 MERKLE TREE IN BLOCKCHAIN DATA STRUCTURE.....	51
FIGURE 4.1 CURRENT PROJECT INFORMATION-SHARING CONCEPT.....	61
FIGURE 4.2 SHARETENDPRO MODEL OVERVIEW.....	64
FIGURE 4.3 ACTORS.....	66
FIGURE 4.4 ACTORS COMPONENT.....	69
FIGURE 4.5 GATEWAY COMPONENT.....	71
FIGURE 4.6 BLOCKCHAIN NETWORK.....	72
FIGURE 4.7 SHARETENDPRO MODEL.....	74
FIGURE 5.1 MODEL DESIGN STRATEGY.....	79
FIGURE 5.2 SHARETENDPRO MODEL ARCHITECTURE.....	80

FIGURE 5.3 USE-CASE DIAGRAM	83
FIGURE 5.4 STATE DIAGRAM.....	85
FIGURE 5.5 OVERVIEW OF THE INFORMATION FLOW	87
FIGURE 5.6 DETAILS OF TRANSACTION T1	90
FIGURE 5.7 DETAILS OF TRANSACTION T2	91
FIGURE 5.8 DETAILS OF TRANSACTION T3	92
FIGURE 5.9 DETAILS OF TRANSACTION T4	94
FIGURE 5.10 DETAILS OF TRANSACTION T5	95
FIGURE 5.11 DETAILS OF TRANSACTION T6	96
FIGURE 5.12 DETAILED INFORMATION FLOW	97
FIGURE 5.13 CLASS DIAGRAM.....	99
FIGURE 5.14 PACKAGE DIAGRAM.....	101
FIGURE 6.1 SHARETENDPRO DEVELOPMENT PROCESS HIGH-LEVEL.....	104
FIGURE 6.2 SHARETENDPRO NETWORK TOPOLOGY DEVELOPMENT PROCESS	106
FIGURE 6.3 CHAINCODE DEVELOPMENT AS THE FOCUS AREA	108
FIGURE 7.1 SCENARIO.....	112
FIGURE 7.2 SHARETENDPRO SOLUTION	113
FIGURE 7.3 HIGH-LEVEL OF THE SHARETENDPRO NETWORK RESULTS	115
FIGURE 7.4 ESTABLISH BLOCKCHAIN NETWORK AS A FOCUS AREA	116
FIGURE 7.5 INTERACTING WITH BLOCKCHAIN NETWORK AS A FOCUS AREA	118
FIGURE 7.6 CREATE TENDER	119
FIGURE 7.7 QUERY TENDER.....	120
FIGURE 7.8 CHANGE TENDER SUPPLIER.....	121
FIGURE 7.9 QUERY CHANGED TENDER REPORT	122
FIGURE 7.10 GET TENDER HISTORY	124

FIGURE A.1 REPRESENTATION OF THE DAG [103]	155
FIGURE A.2 IOTA (TANGLE) ILLUSTRATION [55]	156
FIGURE A.3 REPRESENTATION OF THE HASHGRAPH [113]	157
FIGURE B.1 VISUAL REPRESENTATION OF HARDWARE AND SOFTWARE	164
FIGURE B.2 SHARETENDPRO DEVELOPMENT PROCESS HIGH-LEVEL	165
FIGURE B.3 SHARETENDPRO NETWORK TOPOLOGY DEVELOPMENT PROCESS	167
FIGURE B.4 CRYPTO-CONFIG AS A FOCUS AREA	171
FIGURE B.5 SHARETENDPRO NETWORK TOPOLOGY	172
FIGURE B.6 CRYPTO-CONFIG.YAML FILE STRUCTURE	173
FIGURE B.7 CRYPTO-CONFIG FOLDER	178
FIGURE B.8 CONFIGTX AS A FOCUS AREA	181
FIGURE B.9 MEMBERS OF THE CHANNEL	183
FIGURE B.10 RAFT ORDERING SERVICE.....	186
FIGURE B.11 CHANNEL PROFILES	188
FIGURE B.12 CHANNEL-ARTIFACTS FOLDER	189
FIGURE B.13 DOCKER-COMPOSE AS THE FOCUS AREA	191
FIGURE B.14 CERTIFICATE AUTHORITY SERVICES	192
FIGURE B.15 ORDERING NODE SERVICES	194
FIGURE B.16 COUCHDB SERVICES.....	196
FIGURE B.17 PEER NODE SERVICES	198
FIGURE B.18 CHAINCODE DEVELOPMENT AS THE FOCUS AREA.....	200
FIGURE B.19 TENDER STRUCTURE	201
FIGURE B.20 INVOKE FUNCTIONS OVERVIEW	202
FIGURE B.21 INILEDGER FUNCTION	203
FIGURE B.22 CREATETENDER FUNCTION.....	204

FIGURE B.23 QUERYTENDER FUNCTION205

FIGURE C.1 SHARETENDPRO NETWORK RESULTS OVERVIEW206

FIGURE C.2 DOCKER-COMPOSE SERVICES207

FIGURE C.3 CREATE COMMUNICATION CHANNEL AS THE FOCUS AREA208

FIGURE C.4 CREATE A CHANNEL WITHIN THE BLOCKCHAIN NETWORK209

FIGURE C.5 JOINING PEER NODES TO THE CHANNEL210

FIGURE C.6 UPDATING ANCHOR PEER NODES.....211

FIGURE C.7 DEPLOY SMART-CONTRACT AS THE FOCUS AREA212

FIGURE C.8 INSTALLING CHAINCODE213

FIGURE C.9 APPROVE CHAINCODE214

FIGURE C.10 COMMIT CHAINCODE216

FIGURE C.11 CHANGE TENDER SUPPLIER217

FIGURE C.12 CHANGE TENDER REPORT218

FIGURE C.13 DELETE TENDER219

List of tables

TABLE 3.1 COMPARISONS OF BLOCKCHAIN FRAMEWORKS	55
TABLE 3.2 A COMPARATIVE SURVEY OF THE RELATED WORK.....	58
TABLE 8.1 SUMMARY OF THE BENEFITS AND SHORTCOMINGS OF THIS RESEARCH STUDY	136
TABLE B.1 HARDWARE REQUIREMENTS	159
TABLE B.2 SOFTWARE	160
TABLE 3: COMPARISON OF BLOCKCHAIN FRAMEWORKS	256

List of abbreviations and meaning

AF_N0, AF_N1	Auditor's Firm node 0 and node 1
ACL	Access control list
API	Application programming interface
BCT	Blockchain technology
BTC	Bitcoin
CL	Centralised ledger
DAG	Directed Acyclic Graph
DL	Distributed ledger
DLT	Distributed ledger technology
DM_N0, DM_N1	District Municipality node 0 and 1
ETH	Ether
HLF	Hyperledger-Fabric
ICT	Information and communication technology
IF_N0, IF_N1	Investigator's Firm node 0 and 1
LM_N0, LM_N1	Local Municipality node 0 and node 1
MFMA	Municipal Financial Management Act
MSP	Membership service provider
PBFT	Practical Byzantine Fault Tolerance
PoET	Proof of Elapsed Time
PoS	Proof of Stake
PPPFA	Preferential Procurement Policy Framework Act
PoW	Proof of Work
SAG	South African Government
SALG	South African Local Government
ShareTendPro	Sharing tendering project
SCM	Supply chain management
UML	Unified Modelling Language
XRP	Ripple
YAML	YAML Ain't Markup Language

Declaration

I, Pardon Takalani Ramazhamba, declare that the dissertation titled “Using distributed ledger technology for digital forensic investigation purposes on tendering projects”, which I hereby submit for a Master of Science in Computer Science degree at the University of Pretoria, is my own work. I also declare that this work has not previously been submitted by me for a degree at another University and where secondary material is used, it has been fully acknowledged and referenced in accordance with the University's requirements. I am also aware of the University’s policy and implications regarding plagiarism.

Dedication

I dedicate this work to Ramazhamba S.M and Manaka M.T.S for their endless support and prayers throughout my studies and for teaching me that ‘all things are possible when you put your faith in God’.

Acknowledgment

It would be ungracious of me not to express my appreciation to all the people who helped in making this study possible. I would therefore like to thank the following people and organisations that supported me throughout this study.

- Prof. Hein S. Venter for embarking on this journey with me and believing in me even when I felt defeated at times. I will forever be grateful for his support, guidance, and wisdom in navigating the world of academics.
- The University of Pretoria for giving me the priceless opportunity to conduct this research and gain the knowledge that I have today.
- My RGL, Mr. H. Le Roux (CSIR, Command and Control Systems) for believing in me and supporting this study throughout.
- My mother, S.M. Ramazhamba for her support throughout my entire life.
- My friends for their support and encouragement, especially Miss K.V Mukwena.
- Lastly, I would like to thank CSIR for their financial assistance which was the pillar of this research.

1. Chapter 1: Introduction

1.1 Introduction

We are living in the digital world, whereby majority of people are exposed to digital information. This digital information is driven by the advancement of technologies that are being used daily. Digital information requires an electronic device that has the capabilities of processing and manipulating digital data. These capabilities tend to affect the perception of society as they rely in some way on how we collect, process, analyse, and retrieve data more easily and efficiently [1]. However, some of these capabilities also play a critical role when it comes to innovations, as well as the adoption of information and communication technologies (ICTs). The evolution of these technologies has positively benefited society by providing access to information and improving communication channels. The internet is one of the most used platforms that provide such services. Adversely, there are various risks that come with the usage of the internet. These risks include identity theft, cybercrime, fraud, and many other malicious activities [2] [3]. The most used electronic devices that are targeted by these activities are computers and mobile devices. However, these cyber threats do not stop the adoption of ICT as a tool that aims at enhancing the standard of living [4], because as ICT evolves, new mechanisms are being implemented to address some of these risks.

A lot of organisations also depend on ICT to carry out some of their tasks in an efficient manner. For example, in South Africa, Universities use ICT for registration processes and issuing student results. The South African Revenue Service (SARS) has implemented an ICT platform that allows taxpayers to file their Personal Income Tax Return (ITR12) [5]. Additionally, the South African Home Affairs has also established an online platform that enables citizens to apply for Smart ID Cards and passports [6]. As previously mentioned, computers and mobile devices are the most used electronic devices for such electronic services. Therefore, national departments have also adopted the use of these devices as tools that enable them to perform some of their tasks. However, some of their tasks still use manual processes, where they rely heavily on paperwork to accomplish them. For example, various South African national departments collect tendering information or data using paperwork, which will then be captured and converted into a digital format. Tendering is one of the methods used by organisations to deliver their services. Tendering can also be regarded as a process that might be used by an organisation to procure goods and services using a contractor. A contractor, in this case, can be any registered business or company that offers a certain service to other

organisations. Tendering data is collected whenever there is a call for tendering projects. During the tendering process, some negative or irregular activities that seek to undermine the norms of a tendering system may occur. The problem statement section explores this in detail. This chapter introduces this research study by outlining the problem statement, and motivational details with an aim of contextualising the identified problem. The research objectives and research methodology are set out briefly while highlighting the goal of the study and the research processes that will be taken to address the identified problem. The scope and structure of the study are highlighted to give an overview of each chapter. The last section provides the concluding remarks of this chapter.

1.2 Problem statement

The current South African tendering system still relies heavily on manual processes, which require skilled personnel to deal with forms and administering the entire process [7]. The main reason behind using paperwork is to accommodate all participants including small and new contractors because some of them are unable to share their projects due to various reasons. Some of the reasons are; lack of internet access and the lack of personal computers, which leads to the usage of files to store some of their project information. Project information plays an important role when it comes to awarding a tender to a contractor since it reflects the competency area and project history of that particular contractor. All contractors are required to submit such information when they are applying for a tender. Some of this information will go through a verification process, whereby a referee will then be contacted with regards to a specific item indicated on the documents. A referee, in this case, might be either a client of that contractor or someone who might provide more information or clarity on the contractor's service. Some of the tools that are used to share project information are reports, meetings, presentations, site visits, and other intermediaries such as trusted third parties. Through these processes, information might be altered for corrupt purposes at any given stage.

Therefore, the primary problem of this study is that paperwork is used to share project information, which might contribute towards the illicit altering of information during the process. This might also affect the fairness, transparency, data integrity, and competitiveness of the tendering system. To provide a solution to this problem, the following questions will also be addressed:

- How does the tendering system work in the South African context?
- What are the laws and principles that govern the procurement process?
- Is distributed ledger technology (DLT) a possible solution to the identified problem?
- How does transparency, accountability, and integrity of data or information in a potential solution work and how will it contribute to digital forensic investigations?

The following section provides the significance of this study and how essential the tendering system is in the procurement process.

1.3 Significance of the study

Tendering is an essential procedure for some of the organisational operations as some of these organisations rely heavily on this process to procure goods and services, including information and other inputs. It can also be regarded as the central method used by organs of state to deliver services to their communities to advance social, industrial, or environmental policies [8]. Some of the projects that are distributed using the tendering systems aim to develop and empower the surrounding communities. However, tendering can only be regarded as an essential tool if the procedures and principles that underpin it are adhered to [9]. The mismanagement of the tendering system might lead to interruption of operations, poor product quality, late service delivery, rising costs, and most importantly fraud and corruption [7]. Ngobeni [7] emphasised that the organs of the state are regarded as the country's largest buyer, therefore, there are responsible for ensuring that their tendering system supports and achieves the overall macroeconomic goals. To adhere to this statement, the South African Government (SAG) has constitutionalised the tendering system as one of the methods that seeks to promote social and industrial practices [8]. Therefore, tendering in the SAG should be transparent, cost-effective, competitive, equitable, fair, and free from corruption [7]. The following items depicts some of the requirements that might be adopted to improve the tendering system used by the SAG:

- It should promote the Broad-Based Black Economic Empowerment policy.
- It should allow small and new contractors an opportunity to participate, by reflecting overall participation and contributions.
- It should reduce unethical behaviour such as fraud and corruption.
- It should also increase data integrity, transparency, and auditability.

The following section provides the objectives of the study, which act as the guidelines that steers the entire study.

1.4 Research objectives

There are various ways of providing a solution to a problem, however, there are certain goals that need to be set before attempting to solve a particular problem. Therefore, this study addresses the following objectives:

- i. To investigate how various tendering systems work in South Africa and abroad.
- ii. To investigate whether Blockchain as a technology works and how data transparency and accountability is achieved. Blockchain technology (BCT) can be defined as “peer-to-peer decentralised, DLT that is replicated to all nodes participating in a network” [2].
- iii. To develop a Blockchain prototype that allows organisations to communicate project information securely and efficiently. The proposed prototype might also be used to improve the current tendering project communication in South Africa.
- iv. To investigate how digital forensic might be applied to trace the accountability of the records or data.

The following section provides a solution to how this study is going to achieve its results. The research methodology section explores the adopted methodologies.

1.5 Research Methodology

There are various research methodologies that might be adopted as a guideline used to achieve the desired objectives. However, in this study, the following research methods are adopted namely: design science research (DSR), literature reviewing, modelling, theoretical use-case, prototype, and evaluation methodology as shown in Figure 1.1. This study has adopted the DSR because it is a problem-solving paradigm that seeks to develop or enhance an artifact with an aim of improving the functional performance of the existing tendering system [10]. A literature study on how the tendering system works in South Africa will be conducted as part of an attempt to better understand some of the concepts involved in the tendering system. The literature study will also be used to figure out how BCT works and how it might contribute to a digital forensic investigation. This study will use a wide range of materials. Some of the primary and secondary materials include; the South African legislation, government documents, newspaper articles, journals, textbooks, conference papers as well as materials sourced from the internet.

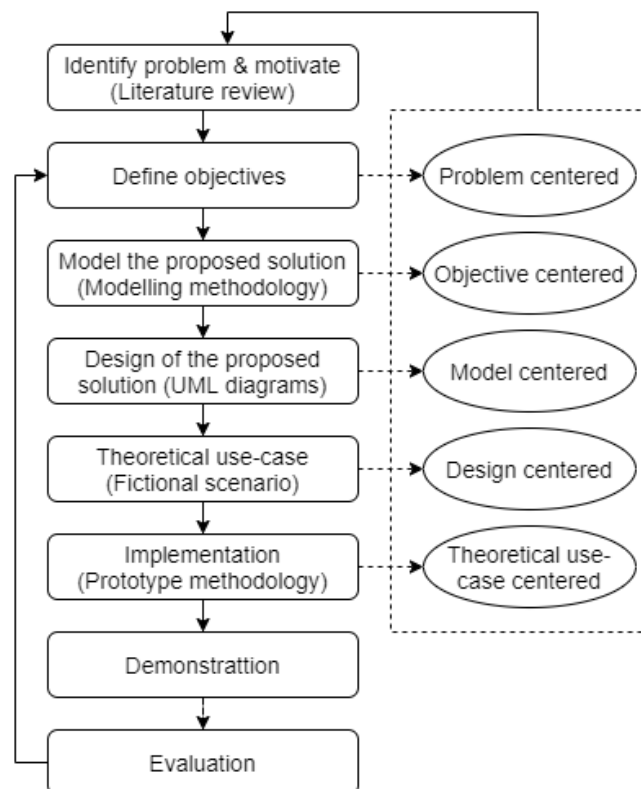


Figure 1.1 Research methodology

After obtaining a holistic idea on how the tendering system work, then this study proposes a model that might be used to share project information securely and efficiently with all the parties that have an interest in the tendering project. Hence, the modelling methodology was adopted to achieve this objective. The modelling methodology is used to model the proposed solution that will enable various parties to interact with the project information stored within the Blockchain implementation or prototype. The proposed model explores how various components interact with the project information to achieve the desired objectives, which is also based on various steps taken during the implementation of the Blockchain prototype. The Unified Modelling Language (UML) diagrams are used to visualise how the proposed model can be used to address the identified problem. Additionally, this study also uses a theoretical use-case to expand the idea behind the proposed model, which is based on a fictional use-case scenario.

Furthermore, the prototype methodology is used to guide the implementation of the desired Blockchain prototype. Note that the prototype methodology also includes the concepts that seeks to test if the proposed solution produces the desired results (e.g., if the system distributes project information among various participants within a network). Lastly, the evaluation is conducted to explore some of the benefits and shortcomings of the proposed solution.

The following section provides the scope of this research study, which also includes the background details of the study area and the role players or communication channels used within that particular area of interest.

1.6 Scope and motivation of the study

The scope of this research is restricted to the South African context, therefore this study does not consider the inclusion of the entire world. However, similar concepts might also be applied nationwide. The SAG and private sector have adopted the use of the tendering system as one of their procurement mechanisms. The focus of this study lies on the tendering system used by the SAG while considering the exclusion of public entities (also known as state-owned entities). These entities are the organisations that have been established through spheres of the government and their main role is to further the programmes of the relevant government department [11] [12]. The SAG is divided into three spheres namely Local, Provincial and National Government. In South Africa, there are nine provinces, which implies that each province has its own Provincial Government. The Provincial Government comprises of a few Local Governments.

The study focuses on the communication that takes place within the Local Government and the Local Government consists of a few municipalities. These municipalities are categorised into three sections namely Metropolitans (Metros), Districts, and Local Municipalities. There are eight Metros in South Africa and the regions that fall outside these Metros are divided into Local Municipalities, while Districts comprise of a number of Local Municipalities. Therefore, this study will focus on the sharing project information between Local and District Municipalities because Metros are regarded as standalone since they report directly to the Provincial Government. However, similar concepts can also be adopted and implemented by these Metros, including the Provincial and National government.

Districts and Local Municipalities share the responsibility of delivering some of the basic services to their surrounding communities. The District Municipality is responsible for overseeing some of the projects executed by their Local Municipalities. Each Local Municipality is divided into wards and each ward consists of a community representative also known as a Ward Councillor. There are various mechanisms that can be used to share project information. However, one of the mechanisms used to share project information with the affected communities is using reports. These communications can take any direction either top-

down or bottom-up mechanism, whereby the bottom side of the communication represents the community while the top part of the communication represents the Municipality. One of the examples of bottom-up communication starts from the ground, whereby the community drafts their demands to their representative, then the representative submits these demands to the Local Municipality. The Local Municipality will either respond by addressing these demands if they fall under their mandate or report the matter to their District. Throughout these processes, there are some protocols that govern the communication channels.

Figure 1.2 represents an overview of the communication channel that is used to share information between District, Local Municipality, and the surrounding communities. This kind of communication is paper-based and it relies heavily on paperwork to share project information. However, this communication channel is also exploited in chapter 4 by proposing a model that might be used to improve such a communication process.

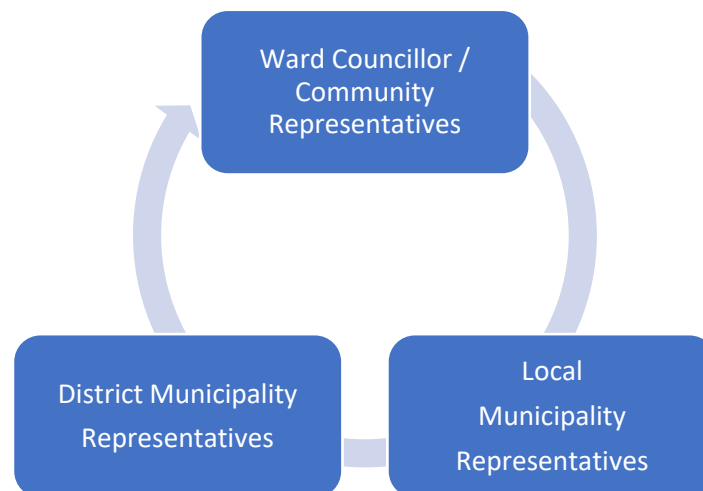


Figure 1.2 Communication concept

In South Africa, there are 44 District Municipalities [13], however, this study has randomly selected one District Municipality as an area of interest that can be used as a proof of concept. One of the main reasons behind selecting a district relies on the sharing of project information with various entities and communities. For instance, Metros share their projects information with their Provincial Government, while several Local Municipalities share that information with their District. Thereafter, the District will then share such information with the Provincial Government. Some of the reasons that contribute towards the selection of the study area include:

- The implicated Corruption Perceptions Index (CPI) 2017 report released by Transparency International, scored South Africa 45 in 2016 and 43 in 2017, which

ranked it 71 out of 180 countries [14]. The scale used by the report is 0 to 100, whereby the score of 0 indicates that a particular country is highly corrupt, while the score of 100 indicates that a particular country is extremely clean [14]. However, the report has indicated that South Africa has dropped 2 points in one year.

- The 2017 Corruption Watch report also indicates that the corruption reports received since 2012 have increased by 25% in 2017 [15]. The report reflected that Gauteng province has the highest number of whistle-blowers with 46%, followed by KwaZulu-Natal province (13%), Western Cape Province (8%), and the province which has the least whistle-blowers is Northern Cape with 2% [15]. The province that constitutes a Metropolitan municipality has at least 5% of whistle-blowers, while four provinces have an equal number of whistle-blowers namely Limpopo, Mpumalanga, Northwest, and Eastern Cape Province.
- The Corruption Watch report also highlights that the Provincial Government has the highest number of whistle-blowers with 30%, followed by the National Government (29%) and Local Government scored a 22%, while Private sectors scored 9% and others scored 10% [15].

This study has randomly selected Capricorn District as an area of interest that can be used to demonstrate how it shares project information with other Local Municipalities, since it consists of five Local Municipalities. These Local Municipalities are Aganang, Blouberg, Lepelle-Nkhumpi, Molemole, and Polokwane [16]. The researcher also acknowledges that these entities use different procurement mechanisms and one of these mechanisms is the use of a tendering system. There are various procedures that underpin the tendering process, therefore some of these processes will not be included during the implementation of the desired tool. Additionally, this study will not consider the inclusion of document sharing, because it aimed at highlighting the importance of securely and efficiently sharing project information, including addressing the issue of data integrity, accountability, auditability, and transparency, while also aiming at eliminating the use of trusted third parties. Note that this study solely focus on proposing a model that can be used to monitor the execution of various tendering projects. Therefore, issues that emanates from the procurement processes such as crafting the tender requirements to favour a particular supplier, tender bidding and awarding of tenders falls outside of the scope of this study. The following chapter, which is Chapter 2 also emphasize this point especially under the importance of monitoring tendering projects since it outlines

that the purpose of this study is to propose a monitoring tool that can be used to provide a continuous feedback by tracking the execution of tendering projects.

The researcher also acknowledges that there are various requirements attached to a particular tendering project. Some of these tendering projects might require confidentiality as part of their requirements, while others might require transparency. This study caters for both of these requirements because it uses a private Blockchain network that can be configured to share confidential information with specific parties. However, note that the examples or use-cases depicted within this study are inclined towards tendering projects that requires transparency since this study aimed at proposing a tool that might be used by various parties to securely and efficiently share project information. Furthermore, note that this study seek to demonstrate a proof of concept that might be used to securely sharing project information with various parties. Therefore, it does not cover all the case studies or tendering projects that can be drawn from the Local Government that might involve other parties that are not discussed in this study.

The following section provides a general overview layout of this research study, and it also provides details on what to expect in each chapter.

1.7 Layout of the study

Chapter 1 provides an introductory section of this research study, and it also highlights the details of the identified problem that this study seeks to address, including the motivational section. Research questions and objectives are discussed briefly. The significance of the study is also highlighted to provide more clarity on how tendering is an essential process when it comes to procurement processes, including the scope of the study.

Chapter 2 outlines the background details of tendering systems in the South African context. These background details include different types of procurement systems and generic processes relating to the tendering system which were discussed. This chapter also highlights some of the South African laws and principles that govern the procurement practices that are found under the Local Government sphere, especially in the Local Municipalities.

Chapter 3 details the background information with regards to the DLT, more especially how Blockchain technologies work and how it achieves its data integrity, transparency, accountability, and auditability in relation to forensic investigations. The fundamental characteristics of Blockchain technologies are also highlighted including the different types of Blockchain technologies and the mechanism used to select the adopted BCT. This chapter also

details some of the core Blockchain frameworks that are available, including selecting a Blockchain framework that might be used to implement the proposed solution.

Chapter 4 addresses the current and proposed project information-sharing concept. Therefore, this chapter seeks to identify the role players of the current project information-sharing concepts (CPISC), including proposing a new concept that might be used for securely sharing project information. Additionally, this chapter also explores the communication channels used by the role players of the CPISC. Thereafter, the proposed concept will then adopt the use of the identified role players and the communication channel used by the CPISC.

This study also adopts the use of BCT as the potential technology capable of addressing the identified problem. Therefore, Hyperledger-Fabric (HLF) is the adopted Blockchain framework used to implement the proposed concept. This chapter further also explores how HLF adds new transactions, including the Business model design and the client application design.

Chapter 5 provides the design of the desired prototype that might be used to share project information by providing an in-depth overview of how it resembles the proposed model. This chapter further explores the system requirements of the desired prototype. The Unified Modelling Language diagrams are used to visualise the interaction of the desired prototype and how users, objects, and the environment interact with each other.

Chapter 6 details the implementation of the proposed solution that might be used to address the identified problem, which states that relying on paperwork to share tendering project information might contribute towards illicit altering of information for fraud and corruption purposes. In other words, this chapter focuses on implementing a prototype that may be used as a blueprint of the desired solution in general because the prototype seeks to demonstrate the proof-of-concept of the proposed solution. Additionally, this chapter addresses the distributed nature of the proposed solution as one of the benefits used to securely share project information among all the participants that have an interest in the tendering project.

Chapter 7 provides a demonstration of the proposed solution, which is a demonstration of the ShareTendPro prototype. In other words, this chapter focuses on demonstrating how the proposed solution or prototype work using various scenarios that seek to portray certain elements or functionalities within the ShareTendPro model.

Chapter 8 focuses on the evaluation of the research with an aim of trying to identify the benefits and shortcomings of this research study. The details contained within this chapter also explores

the pros and cons of the research study because there provides insight of some of the issues identified during either the implementation of the desired solution or the process of conducting this research in general. Some of the items explored within this chapter also seeks to confirm whether the proposed prototype addresses the specified system requirements listed in Chapter 5, since some of these requirements are designed to address some of the objectives of this research study.

Chapter 9 summarises this research study by providing a conclusion, as well as, highlighting some of the future work that can be associated with this research study. Conclusions are based on the information obtained during the evaluation process because there are linked with the identified problem, research questions, and objectives outlined by this research study.

1.8 Conclusion

The purpose and objectives of this chapter were explored in detail, including the adopted research methods, delineation of this study, as well as the significance of this research. The literature study of this research is discussed in the next two chapters, which is Chapter 2 and 3. Chapter 2 focuses on the literature review that seeks to explore how tendering systems work in South African Local Government (SALG), while Chapter 3 focuses on the literature review that seeks to explore the details of the technology that would be adopted by this research study, which the DLT. Therefore, the following chapter, which is Chapter 2 also details the background information of the concepts that are related to procurement proceedings and the principles that govern the tendering system used by the SALG.

2. Chapter 2: Tendering system landscape in the South African Local Government

2.1 Introduction

The previous chapter introduced this research by providing the details that seeks to explore the overview of the study, including identifying the problem this study aims to address as well as the scope of this study. The use of paperwork to communicate project information is the identified problem since it might contribute towards illegal altering of project information due to the paucity of data integrity, transparency, and accountability. The scope of this research study lies in sharing project information in the South African Local government (SALG). Therefore, this chapter details the literature review in relation to sharing of project information as one of the objectives articulated by this study. The researcher also acknowledges that there are various mechanisms that might be used to execute some of these projects. However, this study focuses on projects that are executed using the tendering system, because some processes rely heavily on paperwork.

To achieve this objective, an intensive literature review regarding the tendering system is required, hence this chapter details the background information on the South African tendering system landscape. The remainder of the chapter is structured as follows: the overview of the supply chain management (SCM) used by SALG, different types of procurement systems, and tendering processes are detailed, including the South African principles and legislature that govern these procurement and tendering processes. The importance of tender project monitoring is also presented, followed by the last section which is the conclusion of this chapter.

2.2 Overview of the supply chain management used by SALG

The delineation of this study lies in sharing of project information amongst the South African municipalities which fall under the SALG as highlighted in the previous chapter. Note that the SALG is divided into three categories which are Metro, District, and Local Municipalities as illustrated in the previous chapter or shown in Figure 2.1. District and Local Municipalities share their responsibilities when it comes to executing some of the projects, while Metros are regarded as standalone municipalities since they report directly to the Provincial Government as illustrated in the previous chapter. However, these municipalities use the SCM as a

mechanism that guides the execution of their projects, and the South African National Treasury is responsible for implementing the SCM tool [17]. The SCM tool requires municipalities to have role players who are responsible for executing these projects to address the issue of accountability. Additionally, all these processes are bounded by the legislative frameworks and pillars of procurement [18]. Therefore, Figure 2.1 represents the concepts that interact with the execution of projects and some of these concepts are explained in detail later.

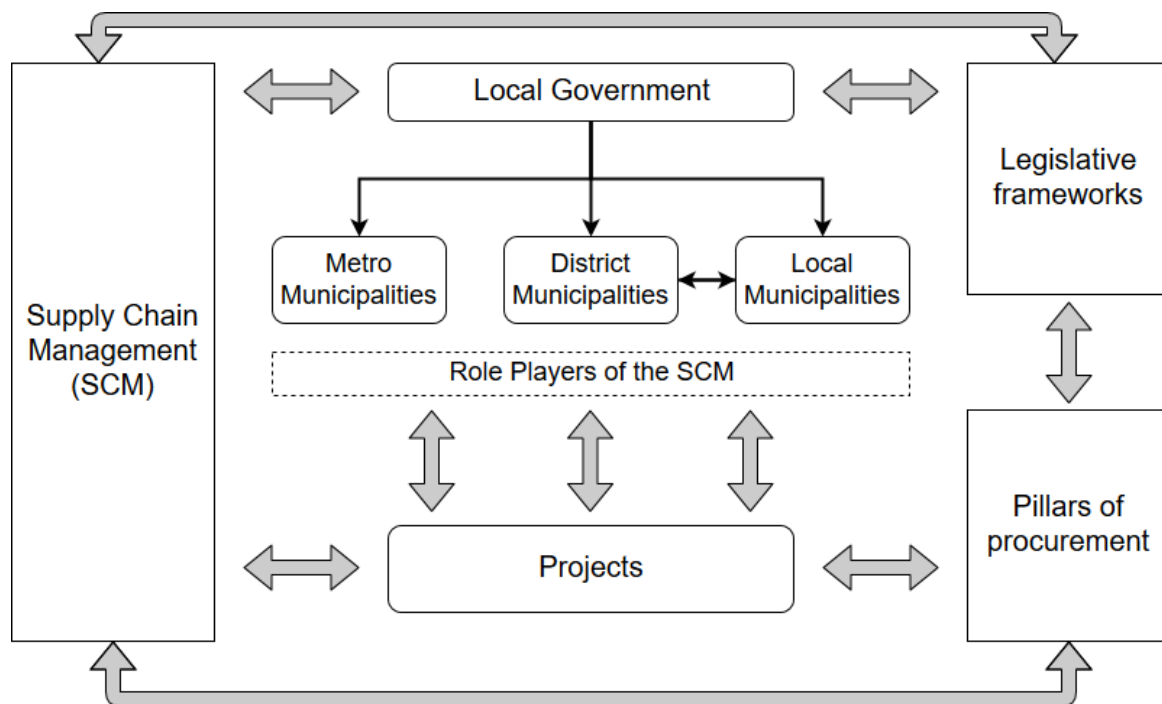


Figure 2.1 Local government projects concept

The SCM can be defined in a various ways, however, this study adopts the definition articulated by the Council of Supply Chain Management Professionals because it defines the SCM as the process that can be used to either plan or manage all the activities associated with procurement or logistics with an aim of coordinating these activities using information systems [19]. This definition emphasises the importance of managing all activities that are involved during the purchase of goods or services from various suppliers. As indicated in Figure 2.2, the SCM consists of eight key processes namely: “*customer relation management*”, “*customer service management*”, “*demand management*”, “*order fulfilment*”, “*manufacturing flow management*”, “*procurement*”, “*product development & commercialisation*”, and “*returns*” [20]. However, this study focuses on one of the major functions of the SCM, which is *procurement*. The main reason behind focusing on *procurement* is that this process requires suppliers to share some of the information with regards to their competency area, including projects history and supporting documents. Some of this information is used for the decision-making process,

especially when it comes to measuring whether a supplier can render a service compared to others. Therefore, it is important to ensure that all the information provided are not tampered with. Figure 2.2 represents the eight key processes of the SCM, including the procurement concept as the focus area.

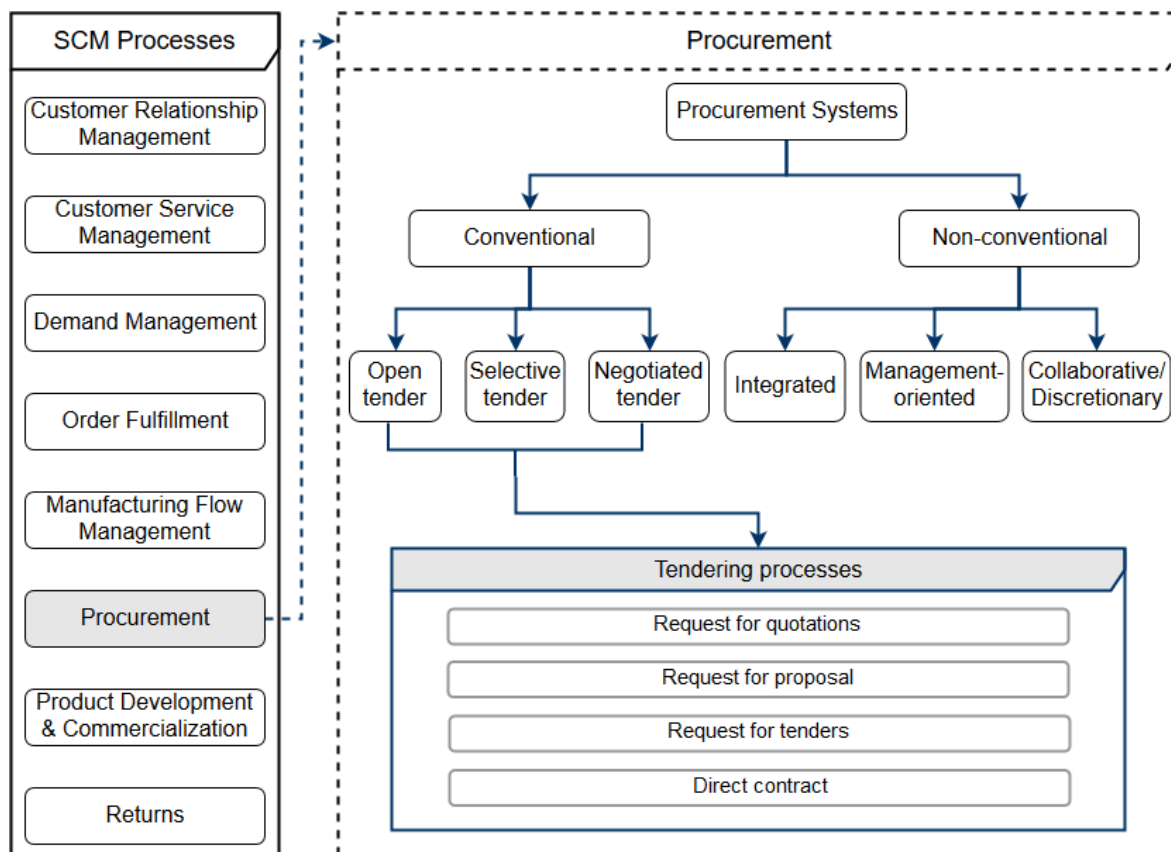


Figure 2.2 SCM processes and Procurement concept

The ISO 10845-1 [21], has defined procurement as the mechanism used by various organisations to create, manage and fulfil their contracts. This definition can be simplified as an act of buying certain goods or services from various suppliers. However, the mechanism used by organs of state to purchase various goods or services from the private sector is regarded as public procurement. Public procurement can be defined as the government administrative activities considered when purchasing certain goods or services required by an organ of the state to perform its functions [22]. These procurement processes are governed by certain policies and procedures, which are adopted from the client or employers' policy framework and this policy framework must map the legislative framework as indicated in Figure 2.1. There are procedures that should be considered before buying any goods or services. The study done by [23] has identified the following procedures: “*identifying goods or services that need to be procured*”, “*selecting procurement strategy*”, “*evaluating tenders or suppliers*”, “*awarding &*

administering tender projects”, and *“assessing the compliance requirements”*. This study focuses on the *“selection of procurement strategy”* since it depends on the goods or services which need to be procured. Therefore, different types of goods or services might require different types of procurement strategies. A procurement strategy might also be used to determine the selection of a procurement method that should be adopted for ensuring the successful execution of the procurement process.

Procurement methods are known by different terms including procurement systems, project approach, and project delivery systems (which are also known as procurement delivery methods) [24]. Therefore, this study adopts the term procurement system as referred to by procurement methods. A procurement system can be referred to as the key instrument that enables clients to create pre-conditions that would be used as a guideline to deliver specific objectives during the procurement process [25]. There are various types of procurement systems that might be utilised to achieve certain objectives. Ngobeni [7], and Thwala et al. [26] classified these procurement systems based on whether they fall under the traditional or non-traditional ways of doing things. However, this study refers to these procurement systems as conventional and non-conventional as opposed to traditional and non-traditional as indicated in Figure 2.2. Therefore, the following sections provide more details regarding each of these classifications.

2.2.1 Conventional

The conventional procurement system refers to the methods used to procure goods and services, which have been used for a long period as the clients’ procurement mechanism. This method allows the client to work with various suppliers such as consultants and contractors. The consultant, in this case, is responsible for designing and documenting the desired project, while the contractor is responsible for executing the designed project. There are various methods that can be used to contract these suppliers. However, the conventional procurement system allows the client to contract these suppliers through the following methods: open tender, selective tender, and negotiated tender [24] as shown in Figure 2.2. The following section discussed these methods in detail.

2.2.1.1 Open tender

The open tender method allows the client to publicly advertise a call for a tender with minimum details describing the requirements and objectives of the tender. It also allows various suppliers

to submit their tender bidding documents to the client. This process requires a minimum amount of money for a supplier to participate in the bidding process [26]. Therefore, there are various situations that favour this type of procurement method, and these situations are:

- When the client wants a highly competitive bidder because the bidding process is open to everyone who wants to participate.
- When the project is not declared as an emergency because all the projects that are declared as an emergency must be executed within a short period to minimise the risks.
- When the client is looking for an experienced bidder for the project since various suppliers are going to submit their bidding documents.

2.2.1.2 Selective tender

The selective tender method allows the client to outsource and shortlist the potential suppliers from either their database or those known to have the capabilities of executing similar projects to submit their updated tender documents for the bidding process. These updated documents consist of supplier's information, which includes information such as competency area, financial standing, and relevant experience. However, this procurement method limits unknown or new suppliers from participating in the bidding process. There are various situations that favour this procurement method, and these situations are:

- When the client does not have enough budget for advertising the tender project since it might be expensive.
- When the clients want to get value for money and better quality for the projects since all suppliers are known to have experience of executing such projects, including reducing the risks of selecting an inexperienced supplier through open tender.
- When the client already has the list of potential suppliers who cannot decline the request, because it might reduce the competence in the bidding process.

2.2.1.3 Negotiated tender

The negotiated tender method allows the client to engage with a supplier regarding a project. If an agreement is reached between these two entities, then that supplier will be required to submit their updated tender documents for the project. There are various situations that favour this type of procurement method, and these situations are:

- When the project is regarded as an emergency since it should be executed urgently and there is no time for advertising it, including looking for a quotation from other suppliers.

- When the client has already built a mutual relationship with some of the suppliers who are known to have the capabilities of executing such projects.
- When the project is declared as national security since these projects are classified, hence specific suppliers are allowed to execute such projects to maintain secrecy.

The following section provides the details of one of the classifications of procurement systems which is a non-conventional procurement system as indicated in Figure 2.2.

2.2.2 Non-conventional

Non-conventional procurement system refers to emerging procurement systems rather than the conventional procurement systems. There are various mechanisms that contribute towards the adoption of these procurement systems and some of these mechanisms include the financial challenges that might arise, the foreseeable complexity of the project, political and social considerations that might affect the procurement system, and the advancement of technology that might be required [26]. However, the non-conventional procurement system allows the client to contract suppliers using the following methods: integrated, management-oriented, and collaborative or discretionary [27] as indicated in Figure 2.2. Therefore, the following section discusses these methods in detail.

2.2.2.1 Integrated

The integrated procurement method allows a supplier to take the initiative of proposing, designing, and executing the project as part of its responsibility on behalf of its client. This method can be divided into three categories namely: “*package deal*”, “*turnkey contract*”, and “*develop and construct systems*” [28]. The “*package deal*” method provides a supplier with the responsibility of proposing, designing, and executing the entire project. The “*turnkey contract*” method enables the supplier to engage with the client with an aim of obtaining an overview of the project and its deliverables before it can be designed or executed. The “*develop and construct*” method allows the supplier to design the project that would be executed by another supplier. There are various situations that favour the three categories that fall under the integrated procurement method and these situations are:

- When the supplier has done thorough research in-line with the proposed project and has all the necessary resources required to execute the project.
- When the client has project ideas and wants to build a mutual relationship with a supplier regarding the execution of some of the project use-cases.

- When the client has the project design in hand that requires a supplier's expertise to execute the project.

2.2.2.2 Management-oriented

The management-oriented method allows the client to work with various suppliers as a joined force with the aim of ensuring that the desired project is executed successfully. One of these suppliers might be responsible for managing or executing the project which was designed by another supplier which is the consultant in this case. These types of projects are mostly found within the construction industry, whereby the execution of that particular project requires experts to examine and oversee the implementation of the project. There are various situations that favour this type of procurement method, and these are:

- When the supplier proposes the design of the project which requires an expert from another supplier to oversee the implementation of the project.
- When the client wants to be fully involved in the execution of the project which is being executed by their supplier with the aim of empowering and up-skilling their employees for similar projects.
- When the proposed project is complex to such an extent that it requires various suppliers to join forces during the execution of the desired project.

2.2.2.3 Collaborative

The collaborative procurement method allows the client to have a framework that will be used to administer the entire project while enabling the administrator to select the most appropriate procurement method from the other two categories, which are integrated and management-oriented [26]. There are various situations that favour this procurement method, and these are:

- When the project execution requires various experts from different organisations to oversee its implementation.
- When the client has adopted the project layout framework from a third party, that requires an external party to execute it.

Thwala et al. [26], emphasised that there is no specific technique that might be adopted to select the most appropriate procurement systems. However, this research study focuses on the conventional procurement systems as depicted in Figure 2.2, because some of its processes require an exchange of information from one organisation to the other. Additionally, the primary objective of this study is based on sharing project information securely and efficiently,

since the current method used relies heavily on paperwork as articulated in Chapter 1. All the procurement systems that are classified under the conventional method are also regarded as tendering systems. Therefore, the following section details the processes that are found in the tendering system.

2.2.3 Stages in the tendering process

There are various processes that need to be considered during the procurement process and these processes are the requests for quotations, requests for proposals, requests for tender and direct contact [7] as indicated in Figure 2.3. This study focuses on the request for tender because it requires organisations to share some of their project information. As illustrated earlier, some of this information is used for decision-making purposes, hence, it is important to ensure that such information is not tampered with. Additionally, the request for tender also consists of various stages namely: “*request for invitation of tenders*”, “*calling for tender bidding*”, “*submission & receiving of tender documents*”, “*opening of tender bidding*”, “*assessment of tenders*”, and “*awarding of tender*” [7] as shown in Figure 2.3. Therefore, the following sections explore these stages in detail.

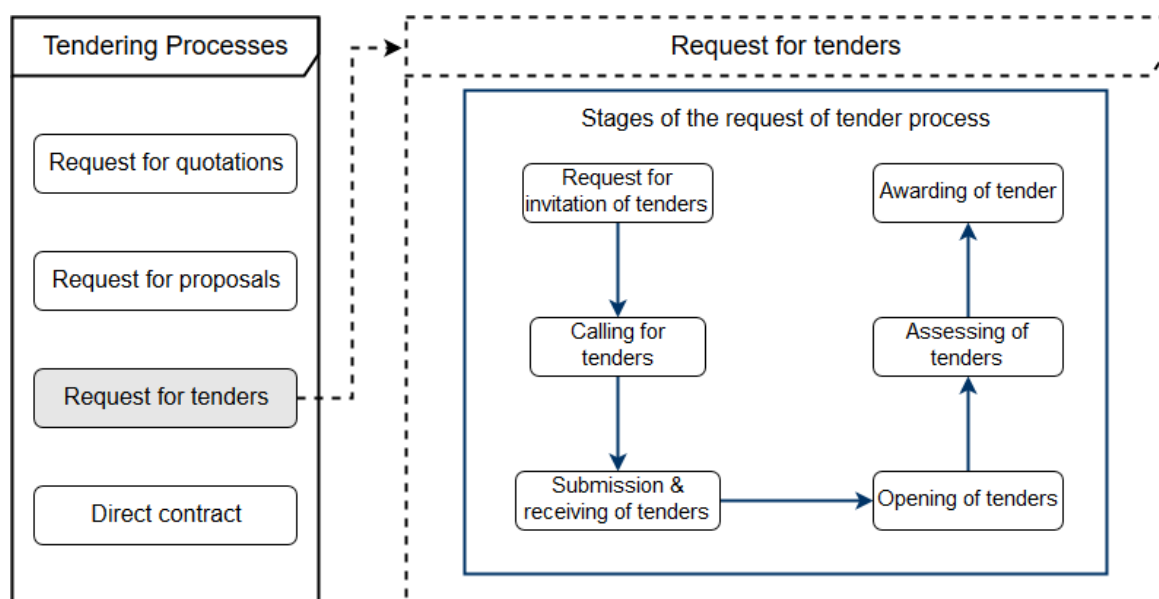


Figure 2.3 Stages of the tendering process

2.2.3.1 Request for invitation of tenders

This stage allows the client to prepare the bidding specifications and requirements of the tender project. The main aim of this information is to describe the required goods or services that need to be procured. However, in the South African Government (SAG), such information should

meet the appropriate standards and general contract conditions set by the South African National Treasury [29]. After compiling all the necessary information and documents, then those details would then be submitted to the procurement unit which will be used for an invitation of tenders. The procurement unit can be referred to as a group of individuals within a specific department (i.e., finance) that are responsible for all the procurement processes within their respective organisation. The model used to guide these procurement processes is referred to as the SCM model [30] [31]. Hence, in South Africa, the procurement unit is also known as the SCM Unit. The SCM Unit will then log a call for tenders if all the pre-requisites are met. The following stage focuses on the calling for tenders.

2.2.3.2 Calling for tenders

This stage allows the SCM Unit to advertise the tender using various mechanisms to accommodate all the participants who want to bid for that tender project. The commonly used boundary for participants lies within the country unless the board decides otherwise [31]. Calling for tender adverts consists of the following information, minimum requirements, description of the service required, contact details for clarification purposes, and the closing date for either submitting queries or relevant documents. There are various platforms that might be used to advertise these tender projects. However, the most commonly used platforms in South Africa are through intermediaries such as the eTender portal and the media (especially newspapers and radio), including organisational websites. The eTender portal was implemented by the South African National Treasury and it allows all the governmental departments and their entities to advertise tenders on their behalf [32]. These tools ensure that the procurement processes are transparent and competitive, intending to get the best value for money by ensuring that they get a competent supplier for the project. The following stage provides details with regard to the submission and receipt of these tender documents.

2.2.3.3 Submission and receipt of tenders

The submission of the tender documents depends on the clients' requirements since some require suppliers to submit their documents either using their tender box or an online system. The commonly used mechanism for such activities in South Africa is using tender boxes, whereby a supplier is required to submit their documents using a sealed envelope to ensure the confidentiality of the information [31]. In some cases, the client might also use an online system that allows suppliers to submit electronic copies of their proposals. These processes are only

valid until the closing date of the advertised tender and all the documents received after the closing date are returned unopened [33]. The following stage details information regarding the opening of tender documents.

2.2.3.4 Open of tenders

There are certain procedures that need to be considered during this stage and these include: the opening of tenders in public, inviting bidders who want to attend, and all late submissions should be returned unopened. The tender information should be announced publicly and added to the official tender registry log for auditing purposes [31]. All the opened tender documents will then undergo an assessment process to compare the feasibility of the proposed bids submitted by various suppliers. The following stage briefly discusses information regarding assessing the submitted tenders.

2.2.3.5 Assessing of tenders

The accounting officer of the department or an entity is required to appoint the bid evaluation committee that will assess all the bids received for the goods or services [30] [31]. The bid evaluation committee will assess all the bids based on the criteria stipulated in the bidding documents. The committee identifies the possible risks that might affect the bidding process and some of these risks include financial standing on the client-side, the availability of adequate facilities that might be required, the capacity required to execute the desired tender project, and the competency area of the supplier [7]. When selecting a supplier, there are minimum criteria that need to be considered for all suppliers which form part of the tender requirements, and the additional criteria include price, quality, availability, and reliability [34]. Some of the tools used by the South African municipalities for assessing the suppliers are the Electronic Tax Clearance system and the SCM system. The ETC system is aimed at reducing fraud and ensuring that the SCM system is not abused, by allowing the state departments and their entities to verify the tax compliance status of the individuals and their organisation, including the broad-based black empowerment equity (BBBEE) status [35]. Some of the information or data used by the ETC system is drawn from the South African Revenue Service, the Company & Intellectual Property Commission, and the government payroll system database [32]. The government payroll system allows government entities to record all their salary payments made using a financial system. The other tool used is referred to as the point system, whereby it collects information from various sources and then produces a report with recommendations.

One of the most commonly used point systems in South Africa is referred to as the government communication and information system (GCIS) [36] [37]. This phase also allows the bid evaluation committee to request more information for clarification purposes regarding certain items stipulated in the tender documents. The following stage provides information regarding the awarding of a tender to a selected supplier.

2.2.3.6 Awarding of tenders

Before awarding any tender, an auditor is required to assess whether the entire procedure has been followed accordingly and non-discriminatory criteria were applied [7] [31]. This process is aimed at reducing the possibility for bidders to contest the award. Moeti et al. [33] emphasise that inviting all the bidders to the tender awarding ceremony reduces the number of contesters since the successful supplier will be announced publicly. However, the successful supplier will be notified using a letter of acceptance, which will then be used as the basis for the placement of orders, administration of contracts, and settlement of disputes [7]. This stage also allows the bid evaluation committee to provide reasons behind their selection, including not considering the lowest supplier.

To ensure that all these procedures and processes are adhered to, there must be laws and principles that will govern them. The following section discusses the South African legislatives that are in-line with the Local government procurement processes, including the role players and the key pillars of the procurement process as presented in Figure 2.1.

2.2.4 Legislative framework

The SALG procurement processes are subjected to a wide range of legislation. However, this section focuses on the legislatives that are associated with the tendering process and these legislations are Constitution, Municipal Finance Management Act (MFMA), and Preferential Procurement Policy Framework Act (PPPFA). Figure 2.4 represents these legislations as highlighted in Figure 2.1. The following sections provide brief details about these legislations.

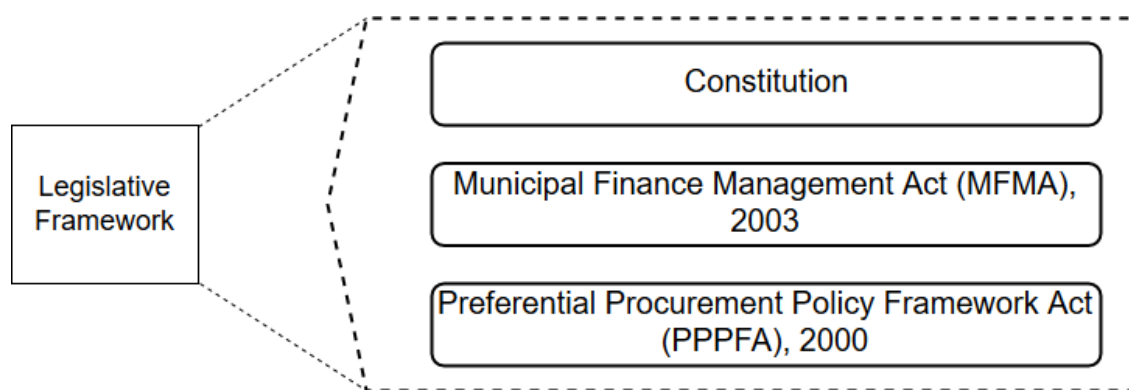


Figure 2.4 Local government procurement legislation

2.2.4.1 Constitution

All the drafted legislation used by various municipalities must be in-line with the South African Constitution. Therefore, the South African Constitution of 1996 (Section 217:1), emphasises that all spheres of the government including institutions identified in the national legislation must ensure that their procurement system is transparent, cost-effective, equitable, fair, and competitive when they contract for goods and services from the private sector [38]. Therefore, to achieve this obligation, it requires the implementation of a system or tool that will reinforce transparency, integrity, and accountability of the data, since these elements might also contribute towards illegal activities such as cyber-crimes, fraud, and corruption, including the misuse of public funds.

2.2.4.2 Municipal Finance Management Act (MFMA), 2003

All the procurement processes used by the SALG are regulated by the MFMA (Act 56 of 2003) [39]. Note that this Act (MFMA of 2003) applies to all the state organs that fall within the SALG and these include all the municipalities and the municipal entities. . All these state organs are regarded as the smallest units used by the SAG to provide or deliver services to the surrounding region or communities. However, Munzhedzi [18] emphasised that all these entities are often characterised by failure to deliver basic services to the communities, financial management challenges, and poor audit outcomes (which in most cases is driven by irregular expenditure, as well as fraud and corruption).

2.2.4.3 Preferential Procurement Policy Framework Act (PPPFA), 2000

This framework provides preferential procurement policies when it comes to selecting the contractor or supplier [40]. Some of its concepts aimed at promoting and supporting the

BBBEE policies, disadvantaged local contractors or suppliers, and section 217(3) of the 1996 Constitution.

All these procurement legislations are incorporated with the core pillars of procurement to ensure that all the procurement processes are adhered to as indicated in Figure 2.1. Therefore, the following section provides the details of these core principles of the procurement process.

2.2.5 Pillars of the procurement process

As indicated in the previous section that the procurement process is not only governed by legislation. However, there are pillars that support the procurement process, and these may differ countrywide. These pillars also act as the core principles that ensure the successful execution of the procurement processes. This implies that if any one of the pillars breaks, the whole process falls apart [17]. The SAG, through the Public Finance Management Act of 1999 has identified five pillars that need to be considered during the procurement process and these are value for money, open and effective competition, ethics and fair dealings, accounting and reporting, and equity [41]. Figure 2.5 represents these pillars of the procurement process as the focus item in Figure 2.1.

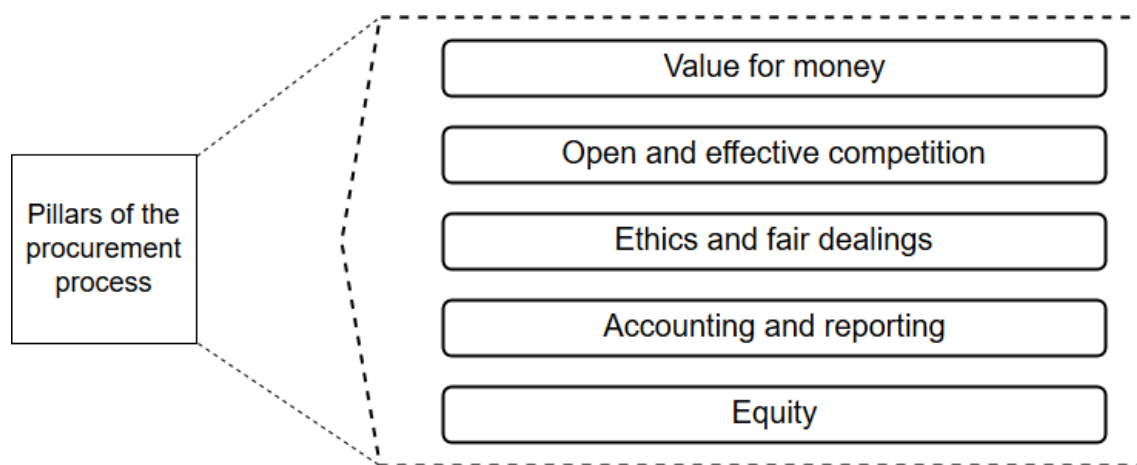


Figure 2.5 Pillars of the procurement process

2.2.5.1 Value for money

This pillar seeks to ensure that all the government entities account for their procurement outcomes. Accepting the lowest bid price is not the only way to get the best value for money because it can also be obtained by assessing the best relevant costs and benefits throughout the procurement cycle [27]. Some of these assessments might contribute towards getting the best value for money includes avoiding unnecessary costs and delays from either the department or

suppliers and monitoring of contracts to ensure that the stated benefits or deliverables are achieved [18].

2.2.5.2 Open and effective competition

This pillar requires government entities to implement laws, policies, practices, and procedures to support and promote their procurement framework [17]. This pillar at ensuring that the procurement systems adopted are transparent, effective to competition, efficacy, and it also subscribes to the guidelines of the PPPFA.

2.2.5.3 Ethics and fair dealings

This pillar allows government entities to comply with the procurement ethical standards and to ensure that they have a fair dealing with all suppliers [27]. However, these entities are also allowed to build mutual trust with their suppliers if all the procurement process dealings are fair, reasonable and integrity [17].

2.2.5.4 Accounting and reporting

This pillar ensures that there are accountability measures in place to make sure that the procurement processes of a particular municipality are open and transparent. This pillar also serves as a mechanism that enables all the individuals who are involved in the procurement process to be held accountable for their plans, actions, and outcomes.

2.2.5.5 Equity

This pillar ensures that the PPPFA policies are adhered to, with an aim of advancing the disadvantaged individuals or tribes by unfair discrimination [18]. It also sought to promote and support all spheres of the industry, especially the small, medium, and large enterprises including creating the opportunity for women as well as promoting local products [17].

These pillars seek to eliminate the possible risks of having loopholes that might contribute towards illegal activities such as fraud and corruption, during the public procurement processes. Therefore, the following section provides the role players of the SCM, with an aim of ensuring accountability during the procurement process.

2.2.6 Role players of the Supply Chain Management

Note that this research study is confined to the SALG as indicated in the previous section. Therefore, this section details the role players of the procurement process that are found under the Local government sphere, and these role players are Accounting Officer, Municipal Council, and the Municipal SCM unit [31]. Figure 2.6 represents these role players as the focus item in Figure 2.1.

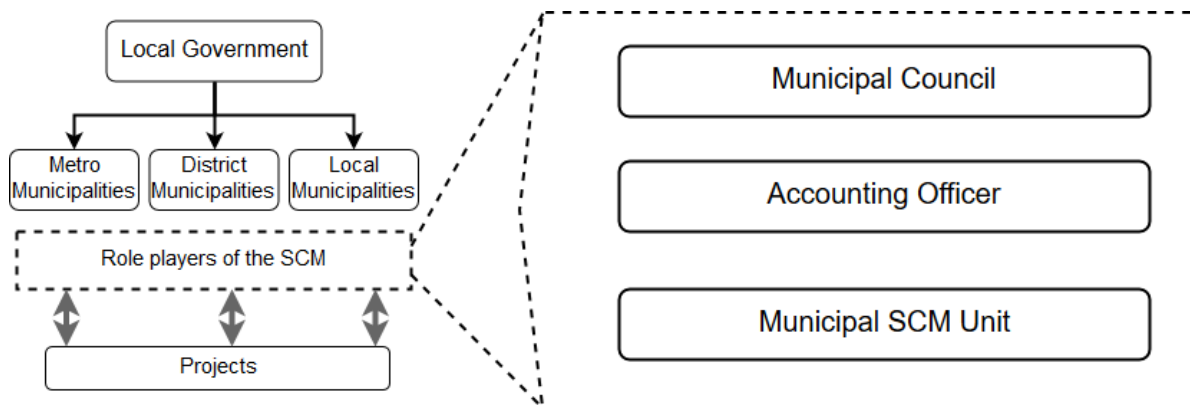


Figure 2.6 SCM role players

2.2.6.1 The Municipal Council

This role player is acknowledged by the MFMA as the highest authority in the municipality [31]. Therefore, it provides clear guidelines when it comes to the responsibilities assigned to the Municipal Council and these responsibilities can be classified into three roles namely: the administrative, accounting officers, and oversight roles. The MFMA also emphasises that councillors are not allowed to participate in the bidding committee, quotations, and contracts including being part of the meetings as an observer.

2.2.6.2 The Accounting Officer

The accounting officers are fully responsible and accountable for all the SCM related expenditures of their municipality as indicated in section 62 and 95 of the MFMA.

2.2.6.3 Municipal SCM Unit

The Municipal Council is required to establish the SCM unit which will be responsible for administering all the SCM procurement processes [31]. This unit ensures that there are clear

lines of authority and accountability while minimising the possible risks that might occur during the procurement procedures.

All these individuals play different roles during the procurement process including decision-making purposes, reporting, assessing, and managing some of the procurement processes. Therefore, the following section provides the importance of monitoring tender projects whilst ensuring the successful execution of projects.

2.3 Importance of monitoring tender projects

Tender projects play an essential role when it comes to stimulating the development of many countries since some of these projects are designed to improve their infrastructure while also empowering the surrounding communities. As indicated in the previous chapter, the government uses tender projects to deliver some of its services and it also invests a huge amount of money for such projects. Therefore, having the legislations, pillars, and role players in-place to govern the procurement processes does not ensure the successful implementation of these projects. However, there are some aspects that need to be considered that contribute to getting the best benefits and value for money out of the projects. These aspects are monitoring and assessment of projects. Otieno [42] distinguishes these aspects as follows: Monitoring of projects as “the process that provides the necessary information and ensures the use of such information by management to assess the effects or impact of the projects”. Assessment of projects is drawn from “the use of data or information generated by the monitoring systems with an aim of analysing the impact of the project trends” [42].

These definitions emphasise that the assessment of projects depends on the monitoring tool since it aimed on ensuring whether the desired objectives have been achieved or not. Therefore, this section focuses on the monitoring of projects because the study focuses on the sharing of project information which falls under the provision of the necessary information for decision-making purposes. Monitoring of projects can also be viewed as a project management tool that focuses on providing continuous feedback on the project implementations. Some of the reasons behind using this aspect as a project management tool include minimising the risk of project failure, promoting the usage of project management techniques, assessing the understanding of the project by stakeholders, including assessing the progress of the project implementation [42]. The most commonly used project monitoring tools include verbal communication,

meetings, reports, and diary notes. However, all these tools have their limitations, and they are also vulnerable to data integrity, transparency and accountability as indicated in chapter 1.

Project monitoring tools act as mechanisms that lubricate the progress of the project with an aim of achieving the desired objectives [42]. Therefore, it is important to adopt an appropriate monitoring tool that will provide the maximum benefits out of the project.

2.4 Conclusion

This chapter outlined the tendering system landscape of the SALG which includes identifying the procurement concepts and processes available. The different types of procurement systems are also articulated including some of the situations that favour these procurement systems. The South African principles and legislation that govern the procurement processes are also highlighted. Finally, the importance of monitoring projects is emphasised, with an aim of highlighting some of the commonly used project monitoring tools available in South Africa.

The following chapter provides the background information of literature on the adopted technology, which is distributed ledger technology.

3. Chapter 3: Distributed ledger technology and security landscape

3.1 Introduction

Chapter 2 explored the background information about the SALG tendering system landscape, which includes information related to different types of procurement processes, legislature, and principles that govern these procurement processes. The role players of the SCM unit and the importance of monitoring tendering projects are briefly explained. Therefore, this chapter discusses the background of the adopted technology used to address the identified problem highlighted in chapter 1. The researcher also acknowledges that there are various technologies that might be used to address the identified problem. However, this study focuses on distributed ledger technology (DLT) since it consists of the features or attributes that are best suited to address the identified problem. Additionally, these features or attributes offered by DLT incorporate the mechanisms of information security, including promoting the need for securely and efficiently sharing of project information.

To achieve the objective of this chapter, a literature review regarding DLTs is required. Hence, this chapter details the background information of the DLT landscape. The details contained within this chapter are structured as follows: background on information security is briefly detailed, including the general information security services which are confidentiality, integrity, availability, identification and authentication, authorisation, and non-repudiation. An overview of the ledger systems is then provided, including the centralised ledger systems and distributed ledger systems. The centralised ledger systems were the first kind of ledger system before distributed ledger systems were developed. All the technologies that are classified under DLTs use a distributed ledger system to share their information. Hence, this chapter also discusses the distributed ledger system in detail, including the consensus algorithms used by the DLTs and the available data structures of these technologies. Additionally, this study favours Blockchain data structure. The determination on when to use Blockchain technology (BCT) is discussed and the Blockchain frameworks that might be used for developing distributed applications. This chapter also summarises the features offered by these Blockchain frameworks and their comparisons. Furthermore, this chapter also outlined some of the related work that can be associated with this study with an aim of providing the gap or ground for this study. In conclusion, the last section summarises the entire chapter with a conclusion.

3.2 Information security

Information security can be defined in different ways. However, Venter and Eloff [43] defined it as the protection of information with an aim of trying to minimise the risks associated with exposing such information to unauthorised parties. There are various information security services that need to be considered to ensure that this information is well protected. These information security services include confidentiality, integrity, availability, identification and authentication, authorisation, and non-repudiation [44]. Additionally, the use of these information security services is to safeguard any information or data that is being transmitted on a network. The following sections discuss these information security services in detail.

3.2.1 Confidentiality

The confidentiality service allows only authorised parties to have access to the data or information shared across the network. The mechanism that can be used to achieve this service is cryptography. Cryptography can be regarded as the science of writing in secret codes since it is aimed at securing communication over an insecure channel [45]. Therefore, the mechanism of converting data into a secret code is referred to as encryption, while the mechanism of converting it back from a secret code is referred to as decryption. Additionally, all the unencrypted information is regarded as plaintext, while the encrypted information is regarded as ciphertext. There are various cryptography algorithms available that might be adopted to either encrypt or decrypt data. However, this research study classifies these cryptography algorithms based on the number of keys it uses during the encryption and decryption mechanism. These classifications can be categorised into three namely: hash functions (one-way encryption), symmetric-key (also known as secret-key), and asymmetric-key (also known as public-key) cryptography. Hash functions cryptography does not use keys, however, it uses a one-way encryption algorithm that produces a hash string. Symmetric-key cryptography uses one key for both encrypting and decrypting data, while asymmetric-key uses two keys, one key is used for encrypting data and the other key is used for decrypting data. Symmetric-key cryptography is primarily used to achieve confidentiality of the information or data and some examples of symmetric-key cryptography algorithms are Advanced Encryption Standards and Data Encryption Standards [45]. The asymmetric-key cryptography enables the node or individual to encrypt data using the receiver's public-key and that particular data can only be decrypted using the receiver's private-key.

3.2.2 Integrity

The integrity service ensures that information can only be modified by authorised parties. There are various mechanisms that might be used to achieve this service. However, for the purpose of this research, hash functions are used to explore how the integrity service is achieved. As illustrated in the previous section that a hash function is one of the cryptography mechanisms that might be used to encrypt and decrypt data or information. Additionally, hash functions use a one-way encryption algorithm that produces a hash string. A hash string is used for data integrity, to check whether that data has been changed or not. The nature of the algorithm allows an arbitrary string of data and transform it to produce a hash value that will be used to represent the actual data. For instance, if a hacker or something manages to change the actual data, then a new hash value will be produced, vice versa. One of the examples of hash function algorithms is Secure Hash Algorithms (SHA), which consists of the following algorithms SHA-1, SHA-2, and SHA-3.

3.2.3 Availability

The availability service ensures that information is available and accessible over a network when it is required. The mechanism that might be used to achieve this service is using a firewall. Venter and Eloff [43] define a firewall as “a software tool installed on a specifically configured computer that serves as a blockade, filter, or bottleneck between a trusted internal network and untrusted external network or internet”. Therefore, the main purpose of using a firewall is to monitor and control all the incoming and outgoing network traffic, with the aim of preventing unauthorised communications.

3.2.4 Identification and authentication

This service ensures that the origin of the information transited on a network can be correctly identified while providing assurance that the identity assigned to that information is not fake. There are various mechanisms that might be used to achieve this service and some of these mechanisms include usernames and passwords, biometrics, and access tokens. All these mechanisms are used to allow or deny access rights.

3.2.5 Authorisation

The authorization service ensures that information is only available to those who have the right to access it. One of the mechanisms that might be used to achieve this service is through an access control list. One of the reasons for using an access control list is to ensure that both users and applications have sufficient rights to perform specific tasks. This implies that the authorisation service seeks to either grant or deny access to certain resources when such request is made by either the user or application.

3.2.6 Non-repudiation

This service ensures that neither the sender nor the intended receiver of the information can deny either sending or receiving it. One of the mechanisms that might be used to achieve the objective of this service is using digital signatures. A digital signature in this case refers to a digital code that can be used to verify the content of the information and the identity of the sender. Additionally, a digital signature is equivalent to a normal signature (handwriting signature) since both are associated with a unique mark of an individual with a body of text, and it must not be forgeable [43]. However, a digital signature can be created using asymmetric-key cryptographic algorithms. As illustrated in the previous section that asymmetric-key cryptography uses two keys (public-key and private-key), one for encrypting and the other one for decrypting data or vice versa. For instance, asymmetric-key cryptography allows data to be encrypted using the sender's private-key and that data can be verified by anyone who has access to the sender's public-key. Therefore, the algorithms presented by Rivest-Shamir-Adleman (RSA) data security are one of the examples of asymmetric-key cryptography algorithms.

The adopted technology, which is DLT relies on asymmetric-key cryptography and hash functions algorithms to secure its information. The following section discusses an overview of the ledger systems, which is one of the key technologies used by DLTs to share information.

3.3 Overview of ledger system

A ledger can be regarded as a principal book or a computer file that can be used to record transactions of specific events. Hence, it can be categorised into two namely:

- Physical ledger (which is in the form of a physical book): This includes physical registers, which are normally used to record students' attendance in schools or visitors' information in an organisation.
- Digital ledger (which is in the form of an electronic file): Software accounting systems, which are normally used by organisations to manage transactions related to their financial events in an electronic manner, such as expenses, invoices, and funding. Another specific example of a digital ledger is a system that is used by banks to produce bank statements for their clients.

As illustrated in chapter 1, this study adopts the use of digital information or data to share project information. This digital data requires electronic devices that have the capability of processing and manipulating digital data. Hence, this study adopts the use of digital ledgers. Furthermore, these digital ledgers evolved with innovation as it moved from being a single digital file (for example a spreadsheet) to a fully-fledged digital ledger system. From this point onwards the researcher simply uses the term ledger system to refer to a fully-fledged digital ledger system. A ledger system can be regarded as a computer program that has the capabilities of recording all the transactions as the state of the database changes. These ledger systems have evolved significantly over the past years from a centralised system to a where it has now become distributed. The following section briefly discusses the centralised ledger (CL) system. Thereafter, a distributed ledger (DL) system is detailed.

3.3.1 Centralised ledger system

The CL systems were the first kind of ledger system used before DL systems because initially there were no computer networks to facilitate DL systems. A CL system functions as a central repository or database that keeps all the records of transactions for an organisation. A central repository is a central place where data is stored and maintained, and it relies on a central authority to process and validate transactions. These records of transactions represent anything that can be regarded as valuable to the organisation, for example, assets, revenues, liabilities, and expenses. Basically, these transactions represent anything that can be stored in a centralised database. Figure 3.1 seeks to illustrate a CL system, whereby all the circles represent the computing nodes that have access to specific data stored in a database. However, user credentials are used to separate the roles in which these nodes play in a network. For instance, a lecturer might use a specific node or computer to enter student marks, while on the other hand, a student might use another node or computer to check their grades or marks. Therefore,

the key highlighted in Figure 3.1 represents the user credentials, whether it is for the lecturer or student, provided by a specific node.

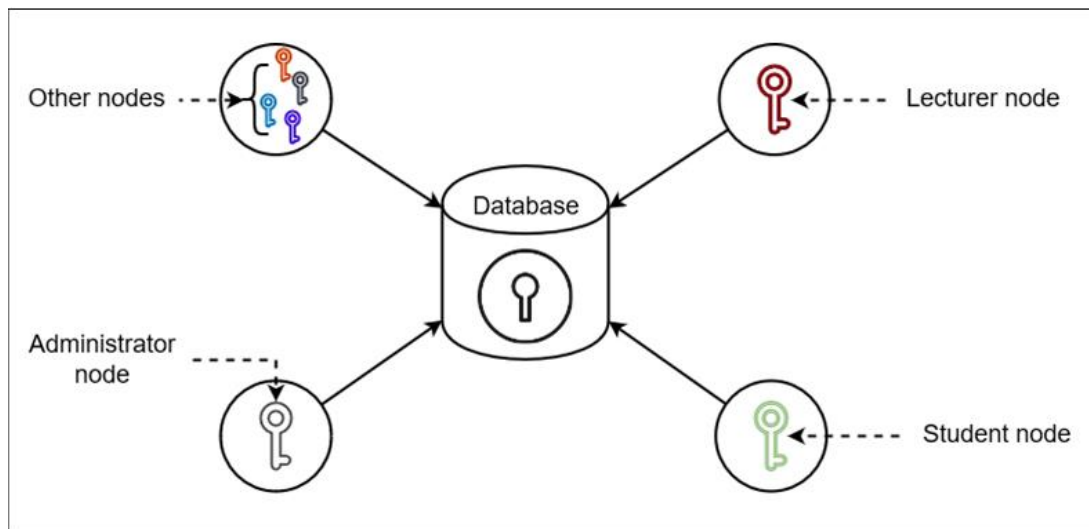


Figure 3.1 Centralized ledger system

Enterprise Resource Planning (ERP) is an exemplary software that is based on CL systems since it collects data from different sources. Some of these sources include purchasing, sales, finance, human resource, and accounting [46]. However, this is an organisational-based system whereby all the information can only be accessed by authorised nodes within the organisational boundaries. A practical example of how a CL system work is the way in which a university's departments use a student management system to share student information with one another. The student management system allows the admin department to be responsible for capturing student information. The finance department is responsible for student finances, while the school's departments are responsible for student marks or grades. Therefore, the student management system in this case act as a CL system for these transactions.

3.3.2 Distributed ledger system

ASTRI [47] defined DLT as a “technology protocol that can be used for developing a replicated and shared ledger system that stores a wide range of assets or transactions in a distributed manner”. A DL system is regarded as a shared ledger system since its records of transactions are maintained across several locations or among multiple nodes, regardless of their geographical location [48]. Basically, this implies that all the nodes that are found within that network have the same copy of the ledger. Hence, a DL system does not consist of a central repository or a single point of failure like a CL system. However, every time when a specific

node in a DL system has made some valid changes on the ledger, those changes are propagated automatically and shared with other nodes that form part of the network. Additionally, this mechanism of sharing information is also aimed at maintaining data integrity across all the nodes within that network. Figure 3.2 seeks to illustrate the DL system, whereby all the circles represent the nodes in the network and the keys on each node represent the cryptographic mechanism used to secure the confidentiality of the ledger.

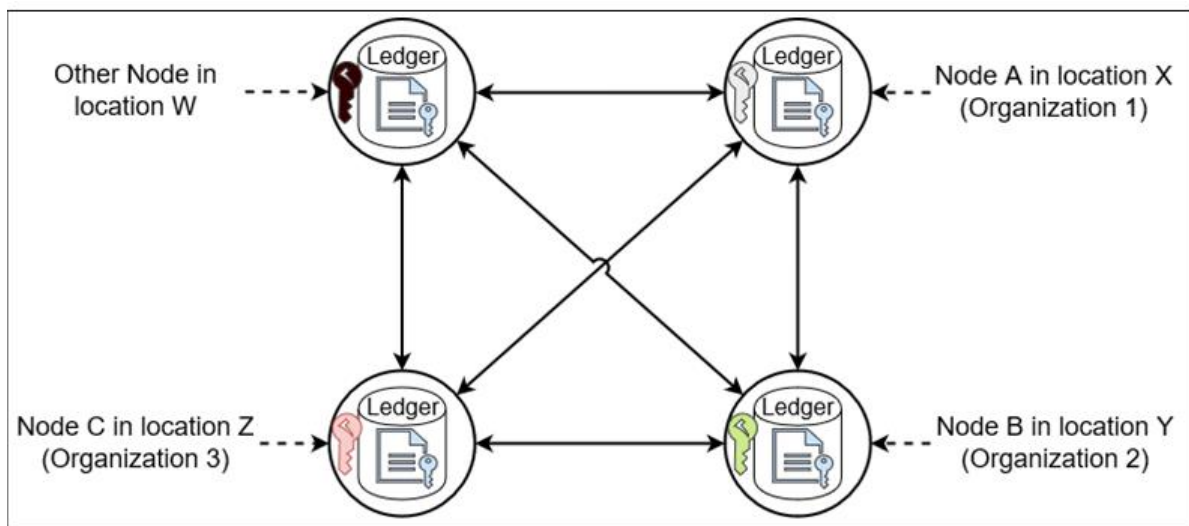


Figure 3.2 Distributed ledger system

A practical example of a DL system is how data centres replicate their data over multiple systems situated in different locations [49]. These data centres are managed and maintained by a single organisation (for example Google data centres). Hence, Google act as a CL system that renders information storage services to its customers since the customers rely on Google to secure their information. However, DLT uses a DL system to record and share all the transactions with all the participants or nodes in a network, unlike Google that stores its information within its centralised data centres with a degree of redundancy. Additionally, once the transaction of storing data has been processed successfully in DLT, that data becomes immutable by default (meaning it cannot be changed) [50]. For instance, if John wants to share a report on project Y with 50 people, John is required to process a single transaction and broadcast it to those 50 people. However, if John accidentally submitted a report for project X instead of project Y, then John is required to process a second transaction to update the mistakes made by the initial transaction. Therefore, DLT will allow a DL system to record both transactions (the initial transaction with errors and the updated transaction with the revised errors) since the data that it records for each transaction, is immutable by default.

Due to the distributed nature of these nodes in a DLT, they need to agree on and check the integrity of the data as mentioned earlier. Therefore, the following section explores how these nodes in a DLT use a consensus algorithm to agree on specific information or transaction.

3.3.3 Consensus algorithms overview

In the field of Computer Science, a consensus algorithm is a process that is used by nodes to agree on sharing a single data value among distributed systems making sure that the shared value is consistent at every node [51]. The purpose of using consensus algorithms is to achieve data integrity amongst the nodes that form part of the network. A practical example of how consensus algorithms work is the one related to a group of 20 people who want to decide on specific projects that might benefit all of them. They might all suggest an idea, but most of these people will favour the idea that will benefit them the most, while others should accept the idea even though it might not be in-line with their objectives.

Typically, in DLT, a consensus algorithm work as follows; Assume that a transaction is shared with 4 nodes in a network, and whilst sharing that transaction, a hacker manages to change a transaction in one of the nodes. Therefore, the network will detect and reject the transaction submitted by the compromised node since 3 nodes will agree on the same transaction, while 1 node will show a different transaction. For example, in an electronic voting environment, the DLT network will represent its votes as follows: 3 nodes accepted, and 1 node rejected. Hence, DLT uses these consensus algorithms as an agreement reached by all the participating members of the network. These consensus algorithms also ensure that only verified transactions can update the ledger system [52]. The verification of these transactions occurs across several nodes, which makes it difficult to tamper with the transaction over its lifecycle because that transaction, which has been compromised or tampered with, can be detected by other nodes in the network. However, a tampered transaction can only be accepted by the network if all the transactions received by the verifying nodes are simultaneously compromised too. This situation will be highly unlikely. Therefore, DLT uses consensus algorithms to ensure that the shared ledgers stored at each different node are the same. This significantly lowers the risks of fraudulent transactions.

There are various consensus algorithms available that can be adopted. However, this study discusses the following consensus algorithms: Proof of Work (PoW), Proof of Stake (PoS), Proof of Elapsed Time (PoET), Tangle, Hashgraph, and Practical Byzantine Fault Tolerance (PBFT) [51] [53]. The main reason for selecting these consensus algorithms is because they

are involved in some of the Blockchain implementations, and they also appear in some of the sections that are going to be discussed. However, this study classifies these consensus algorithms based on how they process new transactions. These classifications are categorised into two categories, namely mining and non-mining consensus algorithms. Therefore, the following section details the mining consensus algorithms. Thereafter, non-mining consensus algorithms are discussed later on.

3.3.3.1 Mining consensus algorithms

The mining consensus algorithms often require powerful computers (also known as miners) to process new transactions. Mining, in this case, refers to a process that is used by miners to add new records of transactions in a distributed network. For the remainder of this chapter, the term “network” is used to simply refer to the term “distributed network”. Miners are often special devices that are designed to offer specific services to ensure that the network function without the need of relying on or having a central authority to handle data integrity and validations. There are various services that are being offered by these miners. However, these miners offer these services using mining pools. A mining pool is a group of miners that organise themselves to form pools where all miners work concurrently to accomplish their tasks [54]. There are various mining pools available that offer a wide range of services in the form of resources. Some of the resources offered by these mining pools include Application-Specific Integrated Circuit, Field-Programmable Gate Array, Central Processing Unit, and Graphics Processing Unit [54].

The services which are rendered by these mining pools are often rewarded with cryptocurrency units. Cryptocurrency is regarded as a digital currency that is governed or regulated by encryption techniques. These rewards are then shared among the miners who contributed during the mining process. There are various processes that need to be considered during the mining process and these processes include [55]:

- *Grouping of unconfirmed transactions into blocks*: this process allows the network to collect all the unconfirmed transactions and group them into blocks based on the timestamp allocated to each transaction. Unconfirmed transactions are all the transactions that are not yet grouped into such time-stamped blocks but already confirmed by the network as valid or successful transactions. A practical example is when John enrolled for course A, and course A consists of three subjects (Maths, Science, and English) and John decided to write two subjects (Maths and Science) and

pass them. Therefore, in this case, John cannot be classified under those who passed or failed the course since one subject is still incomplete. Hence, the completed subjects (Maths and Science) are referred to as unconfirmed transactions for this course even though John has passed them, and they can only fall under confirmed once the third subject is completed since it will determine whether John has failed or passed the course.

- *Assigning a computational puzzle to a block of unconfirmed transactions:* this process allows the network to generate and assign the computational puzzle that requires a computer to solve it by determining the solution of the next block that needs to be added to the network. The notion of solving this computational puzzle is in a form of probability, whereby all the mining devices within that network have the same chance or probability of solving the puzzle. For instance, let's assume that the computational puzzle assigned to this block is in the form of a Sudoku puzzle, whereby miners are required to compete for solving this puzzle as explained next. Note that Sudoku is used as a hypothetical example since the actual puzzle requires miners to generate or create a hash target using a known partial input derived from the latest state of the network. A hash target is used by the network to ensure that the hash of the candidate block that needs to be added falls under a particular threshold. Therefore, miners are required to predict the digital input used to create a hash target that solves the computational puzzle using hash functions. As illustrated earlier on, the hash function uses one-way encryption, hence, miners are required to try many combinations of inputs to create the hash target that solves the puzzle. Furthermore, this process uses up the miners' computer resources such as CPU and memory.
- *Solving a computational puzzle assigned to a particular block:* this process creates a competitive environment whereby miners will have to compete to solve the assigned computational puzzle, which is Sudoku in this case. A practical way of creating a competitive environment is using competition, hence, a competition for the assigned Sudoku puzzle is required. Therefore, the miner that manages to solve the puzzle first wins the competition and, in this regard, the miner wins the opportunity to add the next block by providing a solution to the puzzle. In mining, this solution is regarded as the nonce of the block.
- *Broadcasting the computational puzzle to other nodes for verification:* this process allows other nodes to verify the nonce submitted by a miner by solving the puzzle

themselves. Additionally, once the nonce of that block is verified then it can be used by other nodes to also verify the solution to the assigned puzzle. This process is also explored later on.

- *Broadcasting the block to all the nodes within that network*: this process allows the miners to broadcast the new block that needs to be appended to the network. Additionally, this new block contains the nonce or solution of the computational puzzle assigned to that block, and it can be used by other nodes to verify the solution. If verification fails then the entire block is discarded, which implies that those miners who submitted the nonce will not be rewarded.

However, this study focuses on one of these processes which is “solving the computational puzzle assigned to a block” since it involves the mining consensus algorithms. Additionally, this process consists of procedures that are aimed at identifying the miners who contributed during the mining process. Therefore, the following consensus algorithms PoW, PoS, and PoET are used to identify and reward the miners who contributed during the mining process. Each of these three consensus algorithms is presented in the format as stated by Behrouz [56].

a. Proof of work

Algorithm name: Proof of Work (PoW)

Purpose: to prove that a miner performed a certain task during the mining process

Pre-condition(s): a block of unconfirmed transactions

Postcondition(s): broadcasts special hash code to other nodes in a network

Return: special hash code as a solution to the puzzle

{

Step1: unconfirmed transactions are grouped into a block

Step2: the computational puzzle of the unconfirmed transactions is published to the miners

Step3: miners compete to solve the computational puzzle

Step4: specific miners join forces to solve the computational puzzle by obtaining a special hash code

Step5: those miners broadcast that special hash code for validation

Step6: other miners verify the special hash code and approve or reject as explained before

Step7: the approved special hash code is regarded as a legit solution for the puzzle

Step8: the miners who solved the computational puzzle are rewarded

}

b. Proof of stake

Algorithm name: Proof of stake (PoS)

Purpose: to prove that a miner has invested its stake in that mining process. The stake, in this case, refers to the number of resources each miner has invested in the network. For example, if the network requires 100 megabytes of cloud storage to complete a particular transaction and John has 80 megabytes available, while Peter has 20 megabytes. Therefore, John, in this case, has invested a higher stake in the network compared to Peter.

Pre-conditions: a block of unconfirmed transactions

Postconditions: broadcasts special hash code to other nodes in a network

Return: special hash code as a solution to the puzzle

{

Step1: unconfirmed transactions are grouped into a block

Step2: the system randomly selects the stake to publish the unconfirmed transactions

Step3: the computational puzzle is assigned to the unconfirmed transactions

Step4: miners team up and invest in a particular stake

Step5: miners invested in a selected stake, solved the computational puzzle

Step6: those miners broadcast a special hash code (solution to the computational puzzle) for validation

Step7: other miners verify the special hash code and approve or reject

Step8: the approved special hash code is regarded as a legit solution for the puzzle

Step9: those miners who invested in that particular stake are rewarded

}

c. Proof of elapsed time

Algorithm name: Proof of elapsed time (PoET)

Purpose: to prove that a miner has performed a task during the allocated time frame

Pre-conditions: a block of unconfirmed transactions

Postconditions: broadcasts special hash code solution for the puzzle

Return: special hash code as a solution to the puzzle

{

Step1: unconfirmed transactions are grouped into a block

Step2: the system randomly selects or allocates a time frame for a computational puzzle assigned to the unconfirmed transactions

Step3: the computational puzzle is published to a specific group of miners

Step4: miners join forces to solve a computational puzzle within the allocated time frame

Step5: the miners who solve the puzzle within the allocated time frame broadcast the special hash code as the solution to the puzzle

Step6: other miners verify the special hash code and approve or reject

Step7: the approved special hash code is regarded as a legit solution for the puzzle

Step8: that special hash code is then broadcasted to all the nodes in a network

Step9: those miners that solved the puzzle within their time frame are rewarded

}

These three consensus algorithms are almost similar because they rely on miners to add new or unconfirmed transactions and these miners are rewarded for their services. Using these consensus algorithms, the process of adding transactions often comes with some cost. For instance, the POW consensus algorithm allows miners to compete to solve the puzzle assigned to unconfirmed transactions, and the miners who failed to solve the puzzle often carry the cost that comes with it. This cost can be classified under electricity and processing cost used by that miner when attempting to solve the computational puzzle assigned to a block. The POS consensus algorithm allows miners to invest their resources and the more you invest in a stake, the greater the reward. The PoET consensus algorithm provides a time interval required to solve a puzzle and if a miner fails to solve it during a stipulated time then that miner carries the cost that comes with it. Therefore, the issue of relying on miners and awarding miners whenever a new block of transactions is added to the network is not ideal due to the cost implications attached to it. The issue of cost created an opportunity for new consensus algorithms such as PBFT, Hashgraph, and Tangle. These algorithms seek to eliminate the need for relying on miners to process new transactions. Hence, the following section discusses these new consensus algorithms as non-mining consensus algorithms.

3.3.3.2 Non-mining consensus algorithms

Non-mining consensus algorithms do not rely on miners to process their transactions like mining consensus algorithms. However, the non-mining consensus algorithms rely on validating nodes (also known as validating peers) that check the integrity and validity of the transaction before it can be appended to the business network. The following consensus algorithms Tangle, Hashgraph, and PBFT fall under the non-mining consensus algorithms. These consensus algorithms use a different mechanism to verify and share new transactions with other nodes in a network. Each of these consensus algorithms is presented in the format as stated by Behrouz [56].

a. Tangle

Algorithm name: Tangle

Purpose: to select two trustworthy nodes to validate new transactions that need to be added to the network

Pre-conditions: unconfirmed transactions

Postconditions: broadcasts validated transactions with other nodes

Return: validation status (accept or reject)

{

Step1: a node creates a new transaction

Step2: that node randomly selects two trustworthy nodes as validating peers

Step3: these validating peers are the leaf nodes in a network and cumulative weight is used to select the trustworthy leaf nodes

Step4: validating nodes check the merit of the transaction, and approve/reject

Step5: all the approved transactions are broadcasted to other nodes

Step6: those nodes verify the transaction and update the state of their ledger

}

b. Hashgraph

Algorithm name: Hashgraph

Purpose: to gossip about the new or unconfirmed transactions with neighbouring nodes and to use a virtual voting concept to reach a consensus regarding that transaction. A human analogy on how gossip works are when a lecturer notified a class representative about the new tasks or activities and the students then take that information and share it with all the affected students by word of mouth. A practical example of how gossip work in a hashgraph is explained later on in section A.1 of Appendix A.

Pre-conditions: unconfirmed transactions

Postconditions: other nodes also gossip about these transactions with their neighbouring nodes

Return: gossip status (accept or reject)

{

Step1: node creates a new transaction

Step2: that node randomly selects neighboring nodes to gossip with

Step3: these nodes also randomly select their neighboring nodes to gossip with

Step4: this process continues until all the nodes are informed about the new transaction

Step5: this process of randomly selecting neighbouring nodes is used to broadcast the transactions to all the nodes that form part of the network

Step6: the process uses the push and pulls mechanism to share information with other nodes

Step7: all the nodes virtual votes and orders the transactions based on the time frame it was received by other nodes

Step8: all the nodes come to a consensus using the virtual voting mechanism

}

c. Practical Byzantine Fault Tolerance

Algorithm name: Practical Byzantine Fault Tolerance (PBFT)

Purpose: to ensure that the system network function with a minimum fault tolerance

Pre-conditions: unconfirmed transactions

Postconditions: broadcasts the validated transactions to other nodes in a network

Return: committed transactions that will be added to the network

{

Step1: a node creates a new transaction

Step2: that node uses the pre-prepared concept to broadcast the new transaction with n number of nodes in a network. A pre-prepared concept is a process that is used by a node to acknowledge the transaction received from other nodes and broadcast it to all the neighbouring nodes.

Step3: these nodes also use the prepared concept to broadcast the transaction they have received to all nodes that are found within that network

Step4: the nodes that received the same transaction from both pre-prepare and prepare concept will then commit the new transaction as approved by that node

Step5: these nodes submit their results to the validating peers

Step6: validating peers accept or reject the transaction

Step7: all the accepted transactions are regarded as legit, and the network automatically update the ledger system

}

As illustrated earlier on, DLTs use consensus algorithms to verify and add new transactions, including governing the integrity of the transactions shared amongst all the nodes in a network. Therefore, the following section discusses how DLTs organise or store these transactions. In the field of Computer Science, the mechanism of organising and storing digital data is referred to as data structure. Hence, the following section explores different types of data structures that are used by DLTs.

3.3.4 Distributed ledger technology: the Blockchain data structure

DLT became more prevalent in 2008, after the circulation of a white paper titled “Bitcoin: A Peer-to-Peer Electronic Cash System” authored by Satoshi Nakamoto [57]. The white paper proposed a solution for the financial industry that addresses the issue of double-spending and eliminating the norm of using intermediaries. However, the ideology of the proposed solution existed theoretically [58] [59], until 2009 when the first DLT implementation (Bitcoin system) emerged by Satoshi Nakamoto [57]. The underlying technology used by Satoshi Nakamoto to implement the Bitcoin system was termed “Blockchain” technology. Blockchain refers to the ways in which the proposed system stores and organises its information. The word “Blockchain” is a combination of two words namely “block” and “chain”. Therefore, DLTs use blocks to store their information, and these blocks are linked together to form a chain-like data structure, hence “Blockchain”. As time progresses, similar ways of organising and storing information emerged which lead to the term DLTs as a broad term used to categorise such technologies [48].

DLTs use different types of data structures to organise their information, including the mechanism behind sharing that information with other nodes in a network. Again, note that there are several types of data structures available that might be adopted within the field of Computer Science. Some of these data structures include trees, hashes, arrays, graphs, and linked lists. All these data structures use different mechanisms for collecting elements. However, this study focuses on data structures that are used by DLTs, and these data structures are classified into two categories namely: Blockchain and Directed Acyclic Graph (DAG).

The use of Blockchain data structure in some of the DLTs (i.e., Bitcoin system) raised concerns about the use of miners to process new transactions, including the transactional cost associated with these transactions. Some of these DLTs overcome these concerns by introducing other nodes (computers) that perform the same functions as miners, while other DLTs used an alternative data structure, which is the DAG that does not rely on miners to process new transactions. For instance, HLF is one of the DLTs that uses the same Blockchain data structure that does not rely on miners, since it uses other nodes (also known as peers) to process its transactions [60]. HLF is discussed in detail later under Blockchain frameworks for developing distributed applications section. The DAG data structure is not covered by this section, however, section A.1 of Appendix A explores some of the DLTs that use the DAG data structure to organise and distribute their information with other nodes in a network. Therefore, the remainder of this section's discussion focuses on exploring some of the concepts that use a Blockchain data structure.

A bank account can be used as a practical example that seeks to demonstrate how the DL system uses Blockchain data structure to organise its information. For instance, the most important thing on a bank account is the available balance since it determines the amount you can spend during that time. The available balance or bank balance is derived through credit and debit transactions. Hence, a bank account is comprised of the following components, the one that describes a current state (which is the available bank balance) and the one that describes a set of ordered transactions (which are credits and debits). These ordered transactions of a bank account are presented as a bank statement in this case. However, one can argue that these concepts motivated the first implementation of the DLT (which is Bitcoin system) since it was designed to offer similar services rendered by the banking industry while aiming at eliminating the use of third parties (which are banks) and the reduction of costs (which can be regarded as the bank fee charges and delays during the processing of these transactions). Therefore, the ledger system used by Blockchain technologies (BCTs) consists of a similar concept, however, in BCT the bank balance and the transactions are regarded as a “world state” and a “blockchain” respectively [61]. Each of these components is discussed in detail below

- *World state*: represent information related to the current state of the network (which is like the total net of all the bank accounts of a bank at a given time). The information stored in the “world state” changes frequently (like the changes happening to a bank account due to transactions happening all the time). Therefore, the “world state” acts as a database that stores keys and values of the current state of the network. For instance,

a key, in this case, can be an individual account number and the value associated with that key or account number is the account balance.

- *Blockchain*: represent a set of ordered transactions (which are transactions submitted by different bank accounts) and these transactions are grouped into blocks, which are then connected together to form a data structure that resembles a chain (chain-like). Additionally, the information stored within a Blockchain is immutable, which implies that it cannot be changed or modified. However, note that the Blockchain and its immutability are explored in more detail later on in this section

Figure 3.3 represents a high level of how the two components (“world state” and “blockchain”) organise information within a ledger. As indicated in the previous section, DLTs use the ledger to share their information with other nodes within a network (as shown in Figure 3.2). Therefore, each node contains the same copy of a ledger and this ledger is comprised of two components (“world state” and “blockchain”). The ‘blockchain’ component determines the values stored within the “world state”. For instance, if one can compute the transactions (which occur in the Blockchain data) associated with that bank account then they will obtain an account balance (representing the value within the “world state”) of that particular account.

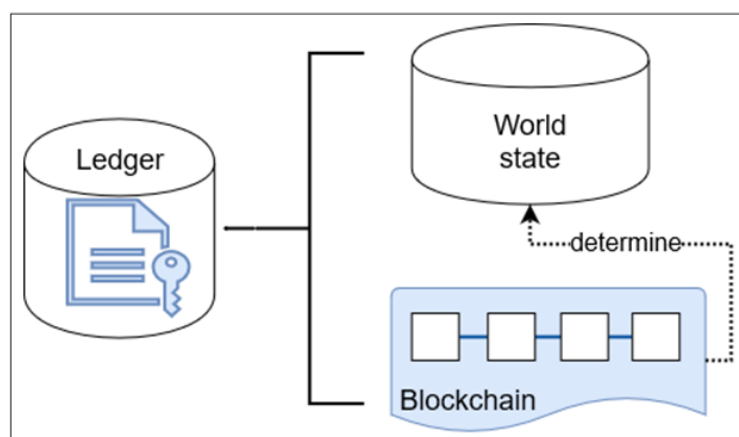


Figure 3.3 Ledger components [61]

Figure 3.4 represents fictional data related to a bank account as an example of how these components (“world state” and a “blockchain”) organises or represent its information. The “world state” changes frequently as the ledger changes from one state to the next. For instance, Figure 3.4 represents the three transitional states of a “world state” (represented by State 0, State 1, and State 2) respectively. All the states represent a total “world state” of 1600. The total “world state” represents the total net of all the account balances (represented by keys and values). The keys are used to uniquely identify each account (which is the account number),

while the values represent the balance of that account. Therefore, to transit from “State 0” to “State 1”, four transactions were submitted (two of them were submitted by Account_1, while Account_2 and Account_3 submitted one transaction each (as shown in the blockchain data between transition states 0 and 1 in Figure 3.4). All these transactions that have resulted in state 1 are stored in the “blockchain” component and the processes below indicate the procedures taken by the network to obtain the new balance after processing the transactions. A to D represents the respective transactions and trans A to trans D represents the processes associated with a transaction as indicated in Figure 3.4. The new balances are stored in the values of the next state, which is “State 1” and the summation of all the new balance (values) constitute the “world state” of “State 1”, which is 1600. Hence, a similar process was followed to obtain the values stored in “State 2”.

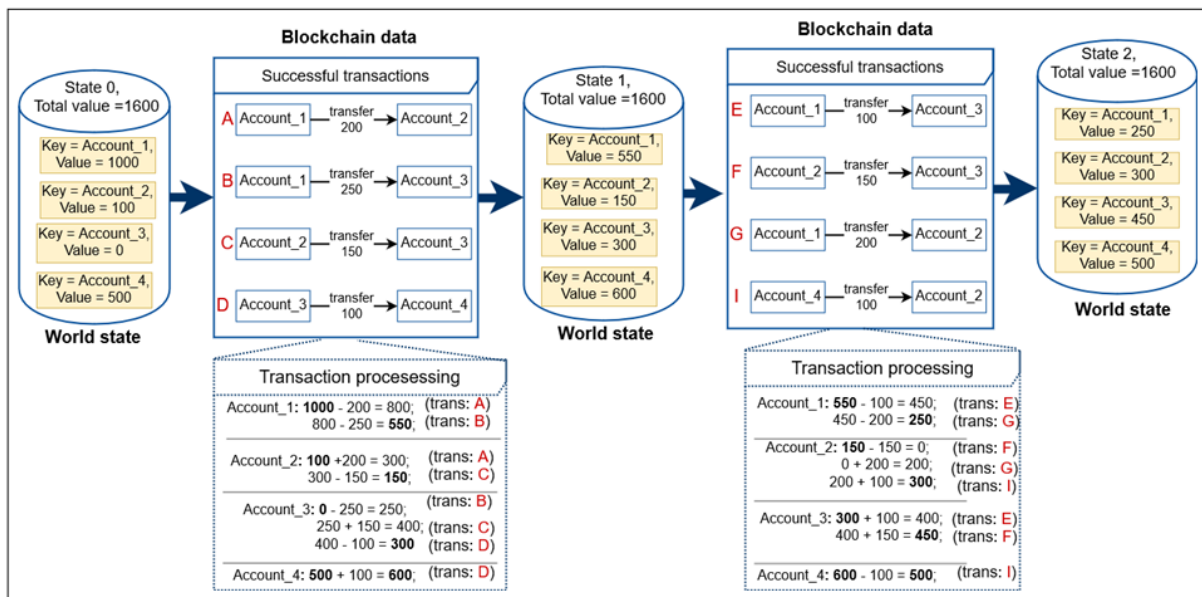


Figure 3.4 Information stored in World state and Blockchain

As indicated earlier on, all the transactions that have resulted in a new state are grouped into blocks and these blocks are appended to a Blockchain. Therefore, Figure 3.5 seeks to represent a “world state” that has transitioned four times, from “State 0” up to “State 3”. All the transactions (i.e. each Blockchain data between states in Figure 3.4) that have resulted in the state are grouped into a block (i.e. block 0, 1, or 2 in Figure 3.4) and each block is then added or appended to a Blockchain (as shown in Figure 3.5) where each block links forward or backward to each respective block, except for the first and last block. Therefore, all the information stored within a Blockchain reflects the entire history of transactions that have

resulted in the current “world state” of the network and this information is immutable as indicated earlier on.

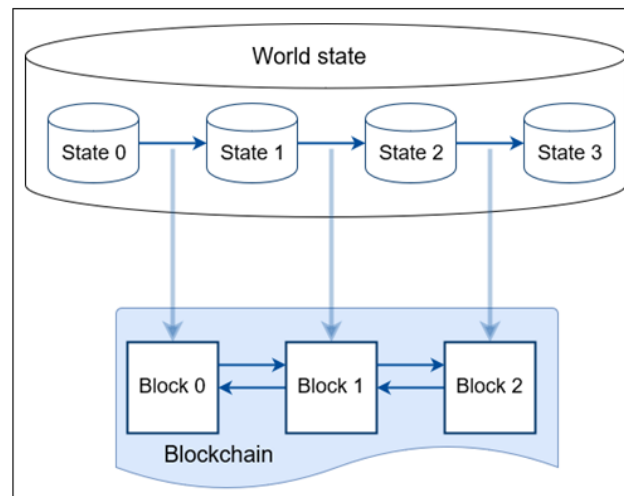


Figure 3.5 Transition of “world state” and Blockchain

Each block within a Blockchain consists of elements that hold or contain specific meta information generated by the network (i.e., block number, timestamp, and hashes of blocks), miners (i.e., the nonce of the assigned computational puzzle), and the transactions (submitted by the users) [62]. The following items describe these elements:

- *Block number*: used for uniquely identifying each block within a Blockchain.
- *Timestamp*: used to capture information related to the date and time when that block was added to a Blockchain.
- *A computed hash value of the previous block*: used to link that block and the previous block. However, each block contains a hash value of the previous block, except the genesis block (i.e., block 0 in Figure 3.5 or Figure 3.6) since it does not have a previous block.
- *Another computed hash value of the next block*: used to link this block and the next block. However, each block contains a hash value of the next block, except the block that represents or reflects the current state of a Blockchain (which is block 2 in Figure 3.5 or block n in Figure 3.6) because it does not have the next block yet.
- *The nonce*: used as a solution to the computational puzzle assigned to that block. The nonce of the block can also be used by any node (as shown in Figure 3.2) within that network to verify the computational puzzle assigned to that block by solving it themselves. However, this element is only used by BCTs that rely on miners to add or append new blocks to a Blockchain, since it requires these miners to solve a

computational puzzle before they can add or append a block to a Blockchain. Additionally, only the miner(s) who manage to solve the puzzle first will be able to add or append new blocks to a Blockchain as indicated in the previous section.

- *Transactions*: represent the actual transactions submitted by users and these transactions are then grouped into a block as illustrated earlier on in Figure 3.4. Therefore, all the transactions within a block are regarded as confirmed transactions because they are already grouped into a block.

Figure 3.6 represents how these elements (block number, timestamp, nonce, a computed hash of the previous block & the next block, and transactions) are incorporated by each block within a Blockchain. Therefore, the links that connect these blocks (i.e., from block 0 up to block 2) form a chain-like data structure (i.e., a doubly-linked list) because it makes it possible to traverse the entire log history that has resulted in the current “world state”. As indicated earlier on, each block consists of a few transactions and these transactions represent an actual transaction or information submitted by a user. Therefore, this study represents a specific transaction located in a particular block as follows:

$$Tm_n, \text{ where } T \text{ represent a specific transaction within a block, } m \text{ is the transaction number, } n \text{ is a block number (1)}$$

For instance, transaction number 2 located in block 1 can be represented as follows:

$$\text{Specific transaction} = T2_1, \text{ since } m=2 \text{ and } n=1 \text{ (2)}$$

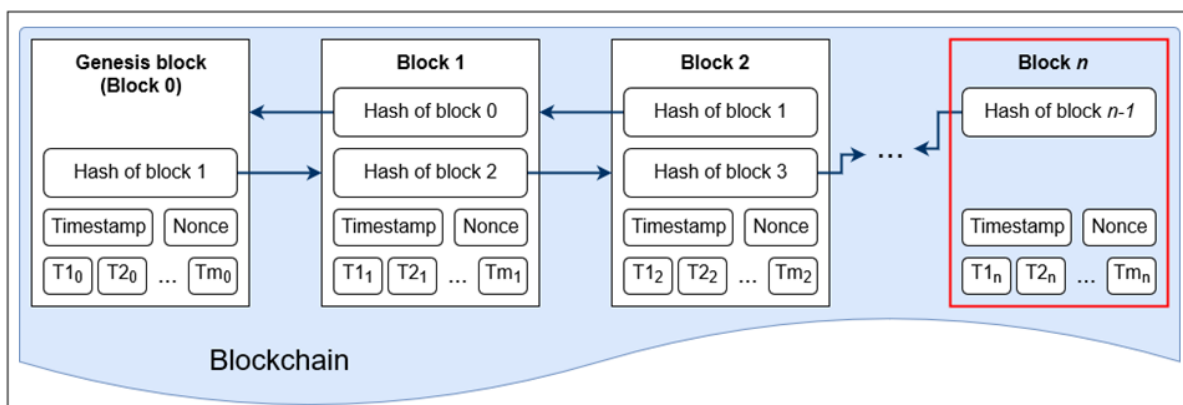


Figure 3.6 Blockchain data structure

The representation of these transactions in this format raised issues related to data integrity because it is difficult to trace or differentiate between a compromised transaction and a legitimate transaction within a block. The integrity deficiencies between blocks can be detected using the hashes in the previous and the next blocks, but not within a block. These issues are

most likely to occur on a block that is about to be appended or added to a Blockchain since once that block is appended, its information becomes immutable by default. This study classifies these integrity issues into two categories namely intentional and unintentional. The intentional issues occur when there is a deliberate attempt to compromise the transaction, while unintentional issues occur naturally due to either a system failure or human error. For instance, intentional issues might occur when a hacker or a miner managed to compromise the transaction within a block that is about to be added or appended to a Blockchain. However, the Blockchain data structure resolved these intentional issues by making use of a Merkle tree [57] as indicated in Figure 3.7, whereby an additional element denoted as a “Merkle root” for the block (i.e., the “Merkle root of block 1” in Figure 3.7), exist for all the transactions within that block. This additional element is also highlighted with green in Figure 3.7. Furthermore, each block within a Blockchain data structure consists of two elements that are used by the Merkle tree, which is the “Merkle root” and the transactions of that block.

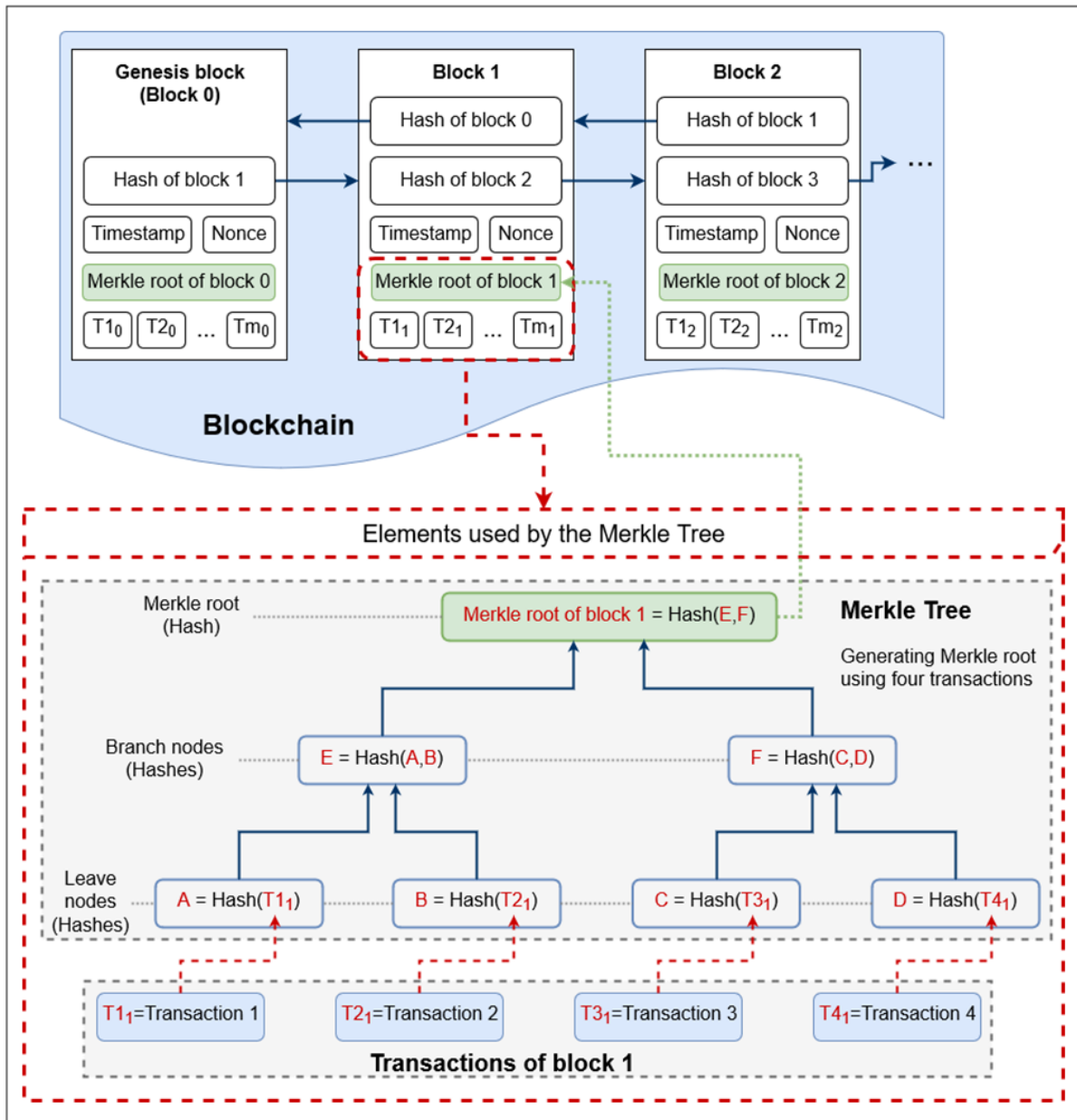


Figure 3.7 Merkle Tree in Blockchain data structure

A Merkle tree is a binary tree in which every leaf node (i.e., A, B, C, or D in Figure 3.7) represents a hash value of the actual data or transactions respectively (i.e., T1₁, T2₁, T3₁, or T4₁ in Figure 3.7). In the field of Computer Science, a binary tree is a data structure in which each node has a maximum of two children (also known as the right and left node), and the node located at the top of the tree is referred to as the root (represented by “Merkle root of block 1” in Figure 3.7). Therefore, four transactions (T1₁, T2₁, T3₁, and T4₁) were used to generate the Merkle root (which is the element at the top of the Merkle tree). Each of these transactions is hashed to produce a unique hash value (represented by A, B, C, or D respectively in Figure 3.7). Two of these hash values (E, F) are generated by combining the hashes of A and B as well

as C and D respectively. Each of these are, in turn, combined and hashed, which produces the hashes for E and F. In other words, hash (A and B) are combined and hashed to produce a new hash which is E and the same process is applied to C and D to produce F. The hashes located in E and F are also referred to as “hashes of hashes” (i.e., hash E is the hash of the hashes of A & B). The new hashes E and F are, in turn, combined and hashed to produce a Merkle root (represented by “Merkle root of block 1” in Figure 3.7), which can be mathematically represented as follows:

$$\begin{aligned} \text{Merkle root of block 1} &= [((T1_1)_\#, (T2_1)_\#)_\#, ((T3_1)_\#, (T4_1)_\#)_\#]_\# \\ &= [(A, B)_\#, (C, D)_\#]_\# \\ &= [E, F]_\# \end{aligned}$$

There are various DLTs that support or use the Blockchain data structure to organise and share its information. Therefore, the following section explores some of the technologies that use Blockchain data structure as Blockchain frameworks for developing distributed applications.

3.3.5 Blockchain frameworks for developing distributed applications

As indicated in the previous section, several DLTs emerged after the Bitcoin system which leads to the implementation of various frameworks. However, all the frameworks that use the Blockchain data structure can be classified into three categories namely: permissionless (also known as public), Public-permissioned, and private-permissioned (also known as private) Blockchain [63].

- Permissionless (public) Blockchain: allows any member of the public to join the network and participate in it.
- Public-permissioned Blockchain: allows any member of the public to verify the records or transactions stored in the network.
- Private-permissioned (private) Blockchain: allows specific members to participate in the network, hence it was designed to support private networks.

The following sections explore some of the most popular Blockchain frameworks that might be used to develop or implement a distributed solution that seeks to address the identified problem in Chapter 1. These Blockchain frameworks are Bitcoin, Ripple, Ethereum, and Hyperledger [47].

3.3.5.1 Bitcoin framework

Bitcoin framework is the first DLT implementation and it was specially designed to support native cryptocurrency known as Bitcoin (BTC) [64]. Additionally, the Bitcoin framework is regarded as a public Blockchain that uses a mining consensus algorithm called PoW [54]. Furthermore, it relies on miners to add new transactions to the network as indicated in the previous section.

3.3.5.2 Ripple framework

The Ripple framework was specially designed for digital currency exchange, remittance, and the real-time gross settlement system [65]. This framework is regarded as an open-source, distributed technology that focuses on payment systems, particularly in banking and finance [66]. It supports a native cryptocurrency known as Ripple (XRP) and it also uses a custom-made consensus algorithm called ripple protocol consensus algorithm [67] [68] [69] [70].

3.3.5.3 Ethereum framework

The Ethereum framework was specially designed to support native cryptocurrency known as Ether (ETH). It also supports smart contract, which is the mechanism used by some of the DLTs to govern the network transactions, without relying on a trusted parties to mitigate the transaction processes. In other words, a smart-contract performs the same functionalities as the general contract signed by two or more parties, since it is aimed at setting up the terms and conditions that enforce the digital trust mechanism [69]. For instance, if John stays in country A and wants to buy a car from Peter, who is in country B, therefore, the smart contract ensures that it enforces trust between them. However, the smart contract used by Ethereum is written in high-level languages (also known as solidity [71]) and compiled by bytecode which requires an Ethereum Virtual Machine to execute it. Furthermore, the Ethereum framework can be regarded as both a private and a public Blockchain that uses PoW and PoS consensus algorithms [72] because it can be configured to support either private or public Blockchain solutions. The use of the PoS algorithm is to overcome some of the issues that come with the use of the PoW algorithm, especially when it comes to power consumption and speed of the network. Ethereum also relies on making use of miners to add new records of transactions to the network. In other words, Ethereum cannot work without any miners.

3.3.5.4 Hyperledger framework

Hyperledger is regarded as an open-source platform hosted by the Linux Foundation and it was established specifically to advance cross-industry Blockchain solutions [73]. Hyperledger does not support native cryptocurrency since it was designed to build a new generation of transactional applications that aimed at establishing transparency, accountability, and trust [73]. It comprises several frameworks, however, this study focuses on the following frameworks supported by Hyperledger: HLF and Hyperledger-sawtooth, because they are specially designed to support cross-industry applications or solutions [72]. HLF supports private Blockchain, and it uses the PBFT consensus algorithm [74], while Hyperledger-sawtooth supports both public and private Blockchain. Hyperledger-sawtooth uses a PoET consensus algorithm that relies on the Ethereum Virtual Machine transaction family [75]. Therefore, this study favours HLF because it supports private Blockchain that consists of a wide range of tested use-cases from different industries compared to Hyperledger-sawtooth. Additionally, HLF supports smart contracts (also known as chaincode in HLF) and it uses validating peers (VPs) to validate its transactions, unlike Hyperledger-sawtooth which relies on an external party (Ethereum) to run some of its modules [76].

All these Blockchain frameworks can be used to develop a distributed solution or application. Therefore, the following section seeks to motivate the selection of the HLF as the selection of a particular Blockchain framework that can be adopted by this study.

3.3.5.5 Selecting a particular Blockchain framework

Table 3.1 compares the above Blockchain frameworks with the aim of selecting a suitable framework and this comparison is based on whether the identified requirements are favourable or unfavourable for this study. However, some of these situations are based on the features or benefits offered by these frameworks.

Table 3.1 Comparisons of Blockchain frameworks

Requirements	Blockchain frameworks [47] [64] [68]			
	Bitcoin [57]	Ripple [66]	Ethereum [77] [78]	HLF [60] [61] [73] [77]
Support cross-industry application development			✓	✓
It must not rely on a native cryptocurrency				✓
Support private Blockchain settings or configurations		✓	✓	✓
Support smart-contract			✓	✓
It must not rely on miners to add new transactions		✓		✓
It must support a data auditing mechanism		✓		✓
Support Blockchain data structure	✓	✓	✓	✓

The comparison of these Blockchain frameworks favours HLF because all the requirements are met. Additionally, all the participating members of HLF are also known (since HLF is a private Blockchain) and such members are, therefore, accountable for their actions. Hence, it can be assumed that all the participants can be trusted with the assigned tasks. This study adopts the use of HLF as the chosen Blockchain framework that will be used to implement the proposed solution, with an aim of addressing the identified problem of relying on paperwork to share project information.

This section has explored all the concepts related to the adopted technology, i.e., Blockchain or HLF framework. This section has also explored concepts related to centralised ledger systems, distributed ledger systems, consensus algorithms, distributed ledger technologies, as well as Blockchain frameworks. Therefore, the following section explores the details of the related work that can be associated with this study.

3.4 Related work

There are several related works that can be associated with this study. Some of these related works tend to focus more on the procurement processes, which includes processes such as applying for a tender, submitting tender documents, tender bidding, and awarding of tenders, including managing tender contracts or projects. Various studies classify the following processes: applying for a tender, submitting tender documents, tender bidding, and awarding of tenders as e-procurement because their information is widely used during the procurement proceedings. The management of the tendering contract or projects tends to come after the e-procurement processes to combat issues that emanate from duplication of contracts or tendering projects.

For instance, the study done by [79] proposed an e-procurement system that can be used to create, publish, bid, and award tendering projects. The proposed system is based on the Ethereum platform, which relies on cryptocurrency or mining algorithms to add new transactions to the main network. The study by [80] also adopted the Ethereum platform to expand the tender bidding concept by including processes such as sharing and verifying tendering information. Additionally, the study done by [81] also adopted a similar approach to expand the tender bidding concept by including processes such as supplier habilitation and delivery verification. However, the model presented by [82] has adopted a different approach or technology solution since it uses the Hyperledger-composer (HLC) tool to implement a prototype that can be used to share data associated with the bidding and awarding of tender projects. Note that HLC makes use of HLF as the underlying Blockchain framework. Additionally, HLC is also regarded as a deprecated tool because none of its maintainers are actively providing support or developing new features for it [83]. The solution presented by [84] also adopted HLF to expand the tender bidding concept by including a mechanism that can be used to monitor the procurement proceedings.

The following studies [85] [86] can be associated with managing tender contracts because they contain some of the elements that seek to eliminate issues that emanate from duplication of contracts or projects, especially in the public sector. Some of the issues that are addressed by these studies relate to data integrity, transparency, and accountability among various individuals that are involved in finalising the procurement contracts. The Mexican Government is one of the countries that has implemented a tool that seeks to manage its procurement

contracts [87], especially managing contracts of the projects that are executed using tendering systems.

The framework presented by [88] focused on how the Blockchain can be used to facilitate data integrity within the document management for construction-related projects. Note that the proposed framework is also based on the Ethereum platform. Additionally, the work presented by [89] explores a framework that can be used to secure tendering records that are highly susceptible to tampering. The study conducted by [90] expanded this ideology by including a concept that seeks to manage construction projects executed by multiple constructors to provide transparency and accountability within the project.

All these studies tend to share tendering project information with a limited number of parties, especially parties that are involved in the procurement processes. A study conducted by [91] presented an open government concept that seeks to promote transparency within the procurement processes and the importance of sharing project information with various parties that have an interest in it. The study by [92] proposed a framework that might be adopted by the South African Government (SAG) to reduce corruption and other issues that emanates from managing procurement contracts. However, this study took a slightly different approach since it proposes a concept that can be used to monitor the tendering project, including sharing project information securely and efficiently among various parties that have an interest in the tendering project.

Table 3.2 summarises the details of the related work by providing a comparative survey that seeks to outline some of the features or issues that were not addressed by these related works. As indicated in Table 3.2, most of the related work make use of the Ethereum platform as their technology solution, while this study adopted the use of HLF. It should be noted that the features in the last four columns of Table 3.2 resemble positive features. For example, the column on “Does not support tender bidding” should be conceived as positive because this study focuses on monitoring the execution of tendering projects rather than processes that fall within e-procurement. The notion of monitoring tendering projects is aimed at ensuring that it is executed successfully and all the parties that are involved during the execution phase account for their action.

Table 3.2 A comparative survey of the related work

Ref.	Key contribution	Features or issues					
		Industry / department	Blockchain Technology	Does not support tender bidding	Support the execution of projects	Does not rely on mining algorithms	Support private Blockchain
[79]	e-tendering system (create, publish, bid, evaluate, & award tender)	Supply chain	Ethereum				
[80]	e-bidding system (sharing & verifying data)	Supply chain	Ethereum				
[84]	Tender bidding and monitoring framework	Supply chain	HLF		√	√	√
[81]	Bidding process, supplier habilitation & delivery verification	Supply chain	Ethereum		√		
[85]	Contract management	Healthcare, Supply chain	Ethereum	√			
[92]	Public procurement framework (contract management)	Supply chain	N/A	√			
[82]	Tendering system (sharing tender data, bidding, & awarding tender)	Supply chain	HLF, HLC			√	√
[89]	Government tender framework	Construction	Ethereum	√	√		
[90]	Managing construction projects	Construction	Ethereum	√	√		
[91]	Government tendering process	Supply chain	Ethereum				
[86]	Contract management (tender bidding, evaluation & awarding)	Supply chain	Ethereum				
[93]	Supply chain conceptual model	Supply chain, logistics process	Ethereum	√			

3.5 Conclusion

This chapter discussed the background information in relation to the DLT landscape and information security, including the general information security services which are confidentiality, integrity, availability, identification and authentication, authorisation, and non-repudiation. An overview with regards to the ledger system was also discussed, including the two types of these ledger systems, which are centralised and DL systems. However, the DL system was explored in detail as the main topic of this chapter which includes information related to the consensus algorithms used by these systems to agree on specific data and the data structures used by these technologies. Additionally, the selection between the Blockchain and DAG data structure was also explored and Blockchain data structure was the chosen data structure adopted by this study. Hence, the technologies that use this Blockchain data structure were also discussed as Blockchain frameworks, and of these Blockchain frameworks, HLF was the selected framework that will be used by this study to implement the proposed solution.

Therefore, the following chapter details the information concerning the proposed model that might be used to address the issue of relying on paperwork to share project information as identified in chapter 1. Additionally, the proposed model is based on the communication concept articulated in Figure 1.2.

4. Chapter 4: A model for securing and distributing tendering project information

4.1 Introduction

The previous two chapters presented a comprehensive background concept which is necessary to proceed with the research for proposing an appropriate model for sharing tender project information. It, therefore, suffices to summarise some of these concepts for the convenience of the reader before introducing the proposed model. The information detailed in Chapters 1 and 2, highlighted the motive that constitutes the focus of this research, while Chapter 3 explored the potential technology that might be used to address the identified problem. The main issue that was identified by this study is the primitive use of paperwork to share project information, even though some of this information plays a critical role especially when it comes to awarding a tender to a Supplier. This problem has exacerbated to an extent that some of these tendering projects' information is being physically or electronically altered for fraud and corruption purposes as highlighted in Chapter 1. The results of this illicit altering of project information might lead to awarding a tendering project to an incompetent Supplier, which leaves some of these tendering projects incomplete or without being implemented.

Therefore, this chapter proposes a model that might be used by the state organs that fall within the South African Local Government (SALG) to share tendering project information securely and efficiently. The remainder of this chapter is structured as follows: the current project information-sharing concept is summarised as part of exploring the stakeholders involved. The scenarios are detailed in the section to follow, which explores how the current project information-sharing concept might be misused, while the fourth section explores the proposed model. Thereafter, the summary of the entire chapter is detailed in the last section by providing a conclusion.

4.2 The current project information-sharing concept

As highlighted in Chapter 1, the South African Local Municipalities are regarded as the smallest unit used by the SALG to deliver some of the basic services to the surrounding communities. Some of these services are delivered using tendering systems as indicated in Chapter 2. These Local Municipalities are responsible for all the tendering projects that fall under their mandate, even though some of these projects are overseen by their District

municipality. The execution of these tendering projects requires these municipalities to contract Suppliers that can execute similar projects. Thereafter, all these municipalities are also required to share some of their project information with all the parties (such as Communities and Investigators) that have a vast interest in the project. The Communities in this case act as the beneficiaries of some of these projects, while the Investigators are responsible for investigating irregularities and illegal activities that might occur during the execution of some of these projects. Therefore, Figure 4.1 illustrates the current tendering project information-sharing concept used by these municipalities to share project information with all the parties that might have an interest in the tendering project.

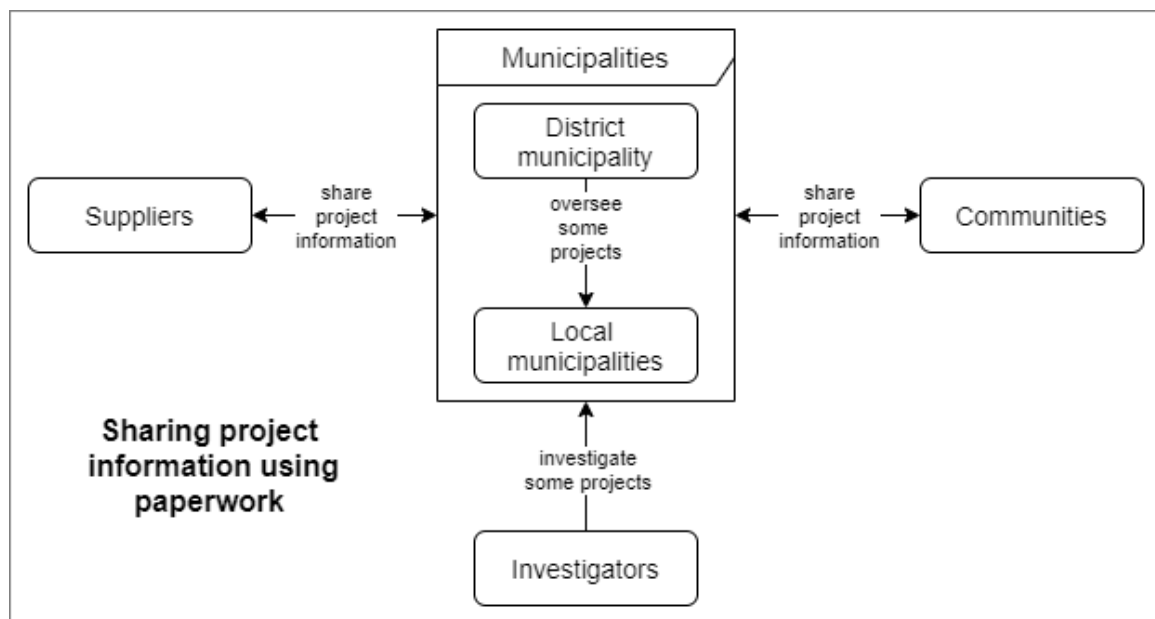


Figure 4.1 Current project information-sharing concept

The use of the current project information-sharing concept has the following security limitations. These are briefly summarised for the purpose of convenience from Chapter 3:

- *Confidentiality*: the current system does not encrypt project information since it uses paperwork to share tendering project information.
- *Integrity*: the legitimacy of the project information might be questionable especially when the softcopy of it must pass through various individuals before the final report can be shared with relevant parties. Additionally, some of the tendering projects might be falsified for corruption purposes.
- *Availability*: the project information can be lost, misplaced, or damaged easily when using paperwork. Additionally, some of the tendering projects might be classified

under undocumented projects due to some of the gaps or missing information in the project report.

- *Identification and authentication*: some of the signatures within the project report might be forged which makes it difficult to identify whether it is authentic or not.
- *Authorisation*: it is difficult to restrict access to project information especially when some of the staff members are sharing an office or have access to the storage facility.
- *Non-repudiation*: sharing project information using paperwork has issues related to the authenticity of the information since the receiver might deny that he has received it, or confirm that he has received it, while in reality, he has not.

All these issues create loopholes for corrupt individuals to exploit the current tendering system used by the South African Local Government (SALG). Therefore, the following section explores the scenarios concerning how the current project information-sharing concept can be misused.

4.3 Scenarios

The following sections discuss these scenarios in detail regarding how the current project information-sharing concept might be misused for fraud and corruption purposes. Therefore, the first scenario is based on a real-world use-case, while the other two are based on fictional use-cases.

4.3.1 Scenario 1

Supplier S and Municipality A might come to an agreement of falsifying a tender project, whereby Supplier S pretend to have rendered services to Municipality A. Thereafter, Municipality A makes use of a “negotiated tender” method to award Supplier S a falsified tender project as an emergency since it allows them to negotiate with a Supplier regarding services as indicated in Chapter 2. Hence, both Supplier S and Municipality A falsify the project reports in a way that reflects that Supplier S has rendered those services of an X amount to Municipality A. One of the examples related to this scenario is a testimony of Mr. X in South Africa under the Commission of Inquiry into State Capture [94]. Mr. X explores how they managed to falsify invoices as the subcontractor of a project without rendering any service to either the main contractor or municipality.

4.3.2 Scenario 2

When Supplier S and a referee within Municipality A have falsified project information to such an extent that Supplier S uses that information to bid for a tender project offered by Municipality B. The Municipality B awards a tender to Supplier S because they have executed similar projects with Municipality A, only to find out later on that Supplier S is incompetent when it comes to executing that project. Therefore, in this case, Municipality B was confident enough to trust that Supplier S is competent since it has included someone from Municipality A, as their referee.

4.3.3 Scenario 3

When Supplier S has been awarded a tender project (assume it is project X) by Municipality A (which is a Local Municipality), that falls under the supervision of Municipality B (which is a District Municipality). Upon reviewing project X reports, Municipality B identifies some of the issues that might lead to project delays or saving costs. Therefore, Municipality B will then notify Municipality A about its concerns in relation to project X, hoping that it will pass these concerns to the relevant Supplier, which is Supplier S. Thus, the communication channel used by the current tendering system restricts the District municipality from communicating directly with the Supplier S because it only plays an oversight role to their Local Municipality.

All these scenarios highlighted some of the issues that need to be addressed. This study focuses on sharing tendering project information because the communication channel used by the current tendering system still relies heavily on paperwork as indicated in Chapter 1. However, some of these issues related to information security, transparency, and accountability within the tendering project can be avoided by using a distributed system that enables actors to share project information securely and efficiently. Therefore, to address these issues, this study proposes a model that might be used by the SALG to share tendering project information. Hence, the following section explores the proposed model in detail.

4.4 Proposed model

This section explores the proposed model that might be used by various municipalities to decentralise and distribute tendering project information securely and transparently. The remainder of this study makes use of the phrase “ShareTendPro model” to refer to the “proposed model”. As illustrated in the previous chapter, Blockchain is one of the potential

technologies that can be used to distribute project information among its actors. Hence, this study adopts HLF (as explained in the previous chapter) as the chosen BCT framework used to implement the proposed solution. Additionally, BCT creates a network that allows various actors to reach a consensus regarding specific information or data as discussed in the previous chapter, under consensus algorithms. To address the above issues, the ShareTendPro model must incorporate the following components namely actors, gateway, and Blockchain network. These components are explored in detail, in later sections, however, these components are briefly explored below for the convenience of the reader to understand the basics concepts behind the ShareTendPro model:

- *Actors*: are the role players of the proposed solution and may, for example, consist of various organisations and their members. A practical example can be Tshwane Municipality (in Pretoria, South Africa) as an organisation and its employees as members.
- *Gateway*: allows actors to interact with the Blockchain network, including the policing mechanism. For instance, a gateway can be used to allow external applications or client devices to submit or retrieve data from the main application (e.g., the use of Google gateway to access geographical locations).
- *Blockchain network*: stores and distributes project information among all the actors that have a vast interest in the project.

Figure 4.2 depicts an overview of how these components (actors, gateway, and Blockchain network) interact with each other.

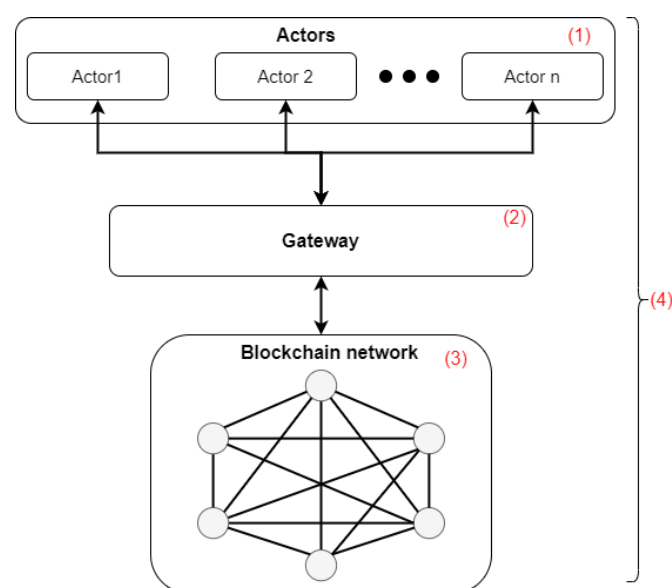


Figure 4.2 ShareTendPro model overview

This study has adopted the following approach to explore how these components work in the proposed solution, as depicted in Figure 4.2:

- 1) Identify the actors of the proposed solution.
- 2) Establish the gateway that will identify and authorise these actors as they interact with project information.
- 3) Establish the ShareTendPro Blockchain network that will store project information securely and efficiently.
- 4) Defining the ShareTendPro model, which is the integration of steps 1 – 3 above.

The following sections outline the details of all these steps that are contained within the adopted approach.

4.4.1 Identifying actors

This study has identified a number of actors that might have an interest in the tendering project information. These actors are classified into two categories namely: main actors and additional actors as shown in Figure 4.3. however, the main actors are all the actors that have direct interaction with project information, while the additional actors are all the actors that have an indirect interaction with project information. The main actors are District municipalities, Local municipalities, Communities, and Suppliers. The additional actors are Auditors and Investigators, even though Investigators might also have some elements that might fall under the main actors, e.g., by visiting the site directly as an observer. Figure 4.3 illustrates the hierarchy of these actors. Additionally, the structuring of the District Municipalities, Local Municipalities, and Communities, is based on how the SAG has divided them into managing or providing basic services to the surrounding communities.

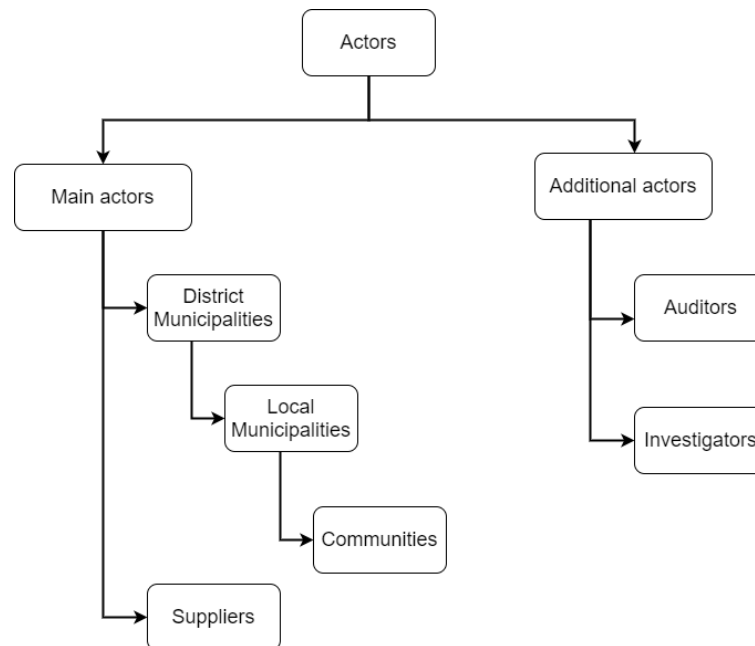


Figure 4.3 Actors

The following sections explore the role played by these actors as they interact with tendering project information.

4.4.1.1 Main actors

As illustrated in Chapter 2, the main actors are District municipalities, Local municipalities, Communities, and Suppliers. The following sections explore the roles played by these main actors as they interact with project information.

4.4.1.1.1 District Municipalities

These municipalities are responsible for rolling out, monitoring, and maintaining tendering projects that fall under their mandate. Additionally, they are also responsible for overseeing some of the projects executed by their Local municipalities as indicated earlier on, including sharing some of the project information with the actors, i.e., Communities, Auditors, and Investigators. Therefore, their role within the proposed solution is to create a contract (by awarding a tender to a Supplier), share project information, and terminating contracts, including accessing project reports of other municipalities.

4.4.1.1.2 Local Municipalities

These municipalities are also responsible for rolling out, monitoring, and maintaining tendering projects that fall under their mandate, including sharing some of these project information with

District municipalities, Communities, Investigators, and Auditors. Their roles within the proposed solution are: to create contracts, share project information, terminating contracts related to specific projects, and accessing project reports of other municipalities.

4.4.1.1.3 Communities

Communities are the beneficiaries of some of these tendering projects because some of them are aimed at developing or empowering the surrounding communities. Additionally, some community members might benefit through employment, while others are stakeholders in some of these projects. However, municipalities are required to share some of their project information with the community representative. Thereafter, the community representative will either share that information with the surrounding communities or organise a community meeting that will provide a detailed report regarding certain projects on behalf of the municipality. Hence, these community representatives play an important role since they act as an intermediary between the community and municipalities. These communities will play the following role within the proposed solution: to access and share project information.

4.4.1.1.4 Suppliers

Suppliers are the private organisations that seek to provide some of these services on behalf of the SALG, which is District and Local municipalities in this case. As indicated earlier on, these municipalities roll out tendering projects that require these Suppliers to compete using the tender bidding process. Additionally, this process allows the municipality to select a Supplier that will render certain services on its behalf. These Suppliers are at the forefront of executing all the tendering projects. All these Suppliers report directly to the municipality which awarded them the tender. Therefore, the role of these Suppliers within the proposed solution is to share project information.

4.4.1.2 Additional actors

These additional actors play an essential role when it comes to ensuring that these municipalities are accountable for their actions, especially when it comes to identifying activities related to irregular expenditures, fraud, and corruption within these projects. The following sections explore the role played by these additional actors, which are Auditors and Investigators.

4.4.1.2.1 Auditors

Auditors are responsible for ensuring that municipalities account for their actions by auditing their financial expenditures to check for irregularities and misuse of public funds. To achieve this objective, municipalities are required to submit their financial reports to the Auditors for auditing. In some cases, the Auditors might require additional tendering project information from these municipalities to elaborate or clarify some of their expenditures concerning a specific project. Therefore, Auditors will play the following roles within the proposed solution: accessing project reports and sharing the finding of the audit reports. This study acknowledges that Auditors randomly select sample payments or claims of various projects, hence their reports would be considered as an optional within the proposed model since not all projects would be audited by an Auditor. Additionally, note that Auditors might use these projects reports to verify whether they accurately reflect the reality of some of the claims or payments associated with a particular tendering project. Hence, these project reports might be used to compare the source records filed by the financial department of a particular municipality.

4.4.1.2.2 Investigators

Investigators are responsible for gathering all the possible evidence that identifies the occurrence of illegal activities within a tendering project. Additionally, the process of gathering this evidence aims to identify the individuals who might be involved in such activities, including presenting such evidence to legal authorities or court. Hence, Investigators play the following roles within the proposed model: accessing project reports and sharing the preliminary findings of the investigation to recoup some of the amount lost due to fraud and corruption activities. This study also acknowledges that not all projects will undergo an investigation, their reports are also considered as an optional within the proposed solution.

Figure 4.4 depicts the interaction of the following actors with the tendering project information: Local Municipalities, District Municipality, Auditors, Investigators, Communities, and Suppliers.

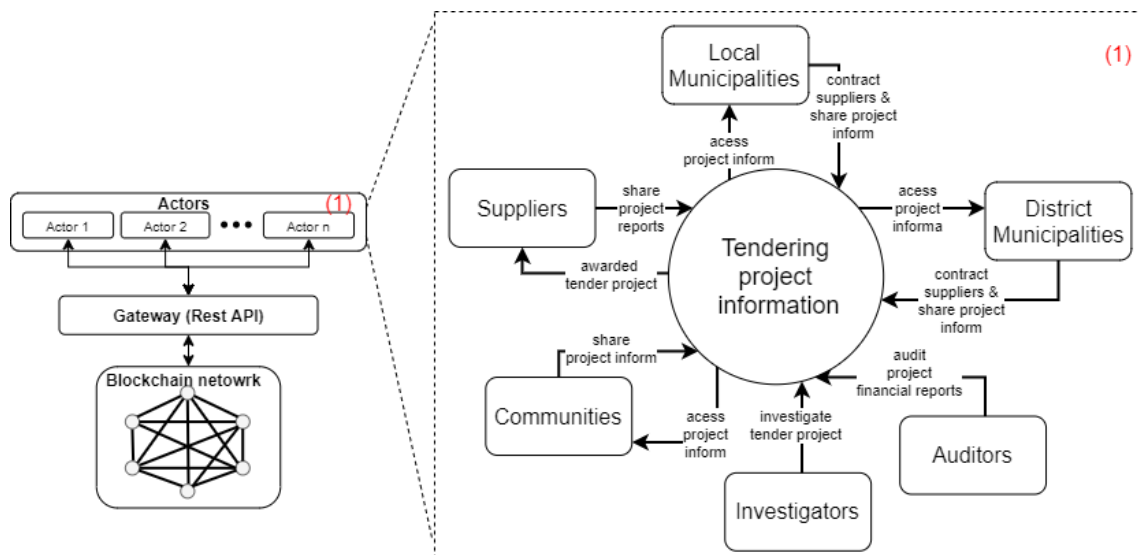


Figure 4.4 Actors component

This section has explored the main and additional actors of the proposed model, even though the reports generated by the additional actors might be regarded as optional since not all the projects will undergo either auditing or investigation. Therefore, the following section explores the gateway mechanism used by these actors to interact with the tendering project information using the proposed solution.

4.4.2 Establishing the gateway

The gateway component serves as the policing mechanism that allows these actors to connect and interact with the network of the proposed solution. This component also plays a critical role when it comes to achieving the following mechanisms namely: authorisation, identification, and authentication of these actors. The authorisation mechanism can be achieved using an access control list, while the identification and authentication mechanism can be achieved using a username and password. However, BCT also makes use of asymmetric-key cryptography (which consists of two keys public-key and private-key) to achieve these. Therefore, all these actors are assigned these two keys and use them whenever they interact with Blockchain data. BCT allows these actors to share their public keys for them to share a secret key that is used to preserve confidentiality, i.e., to secure the communication channel as they share confidential project information with each other. For instance, Municipality A and Supplier S will exchange a secret key whenever they share project information. This allows Municipality A and Supplier S to create a confidential contract that can only be viewed or decrypted by themselves.

The BCT also uses the gateway component to separate the roles played by various actors within the network. The HLF framework achieves this by using the following mechanisms: REST-API, access control list, and secure communication channels. The following items explore these mechanisms in detail:

- *REST-API*: allows various actors to use an application programming interface (API) to interact with the Blockchain network. In HLF, a Composer REST server can be used to generate a REST-API (also known as Swagger REST endpoint API) from a deployed HLF network [95]. In other words, this process exposes the deployed network as a REST-API that allows authenticated actors to interact with the Blockchain data using queries. All the transactions submitted through the REST-API will be assigned an HTTP request operation which either creates, reads, updates or deletes information or data stored within the network. In addition, all these transactions will be signed by a digital certificate to preserve non-repudiation.
- *Access control list (ACL)*: manages the access rights of all the authorised actors as they interact with the Blockchain data. These access rights can be categorised into two namely read and write access. In other words, the ACL consists of the listed actors who either have read or write access, or have both read and write access rights, or might not have any access at all. For instance, Communities, Suppliers, Auditors, and Investigators are not allowed to write or create contract details, however, they are allowed to read it or view some of the details contained within it.
- *Secure communication channels*: allows a specific group of actors to secretly share project information. For instance, a channel might be created for certain Local Municipalities that fall under a specific District to share project information since some of their tendering projects are overseen by a particular District Municipality.

Figure 4.5 represents the above mechanisms, i.e., REST-API, ACL, and secure communication channels, used by the proposed solution to manage the identities of various actors, including providing them with access to the Blockchain network.

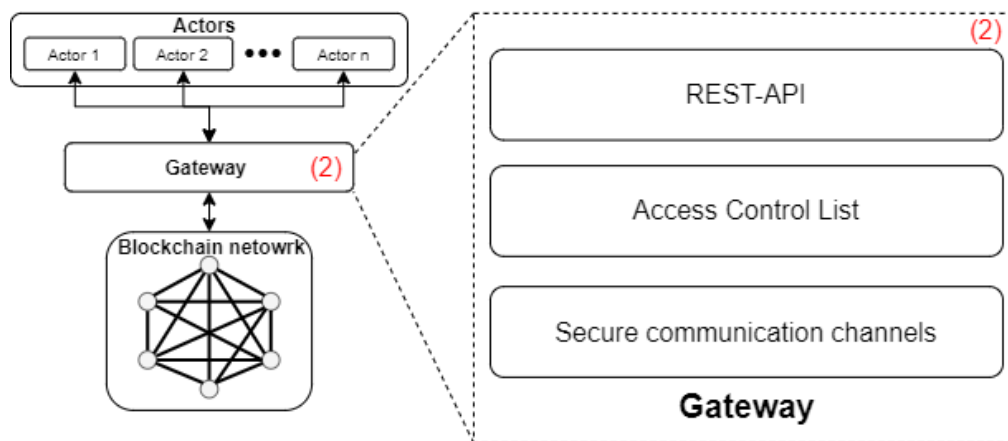


Figure 4.5 Gateway component

The following section explores how the Blockchain network component work, including how project information is distributed among various nodes within the network.

4.4.3 Establishing the ShareTendPro Blockchain network

The Blockchain network component focuses on how the project information is stored and distributed among various nodes within the network. In other words, the Blockchain network component deals with the operational concept or logic behind storing and sharing project information with all the actors that have a vast interest in that tendering project. The Blockchain network component achieves this by allowing all the authorised actors to submit project information as transactions. However, all these transactions should meet specific requirements associated with them. As indicated in the previous chapter, HLF makes use of a chaincode to govern all the transactions within the network. Hence, it consists of predefined conditions associated with each transaction. All the transactions that do not meet their requirements are discarded or declared as rejected.

Therefore, all the accepted transactions are forwarded to the ordering service to be grouped in an orderly manner. The ordering service is responsible for collecting all the accepted transactions within the Blockchain network and grouping them into blocks. Thereafter, all the accepted transactions (blocks) are then shared with all the nodes from part of the Blockchain network as indicated in the previous chapter. The Blockchain network component achieves this by using a distributed ledger system that allows it to distribute these blocks of transactions to various nodes as shown in Figure 3.2. However, each node will then make use of the chaincode (smart contract) to verify these ordered transactions before appending them to the ledger. Once this process is complete and all the nodes have appended the new transactions to their ledger,

then all the actors who have a vast interest in that tender project will now have access to the updated project information or reports. Figure 4.6 depicts how the Blockchain network component distributes project information among various nodes or actors. Part A of Figure 4.6 represents the information flow, while part B represents the distributed nature of the nodes or actors as they share project information.

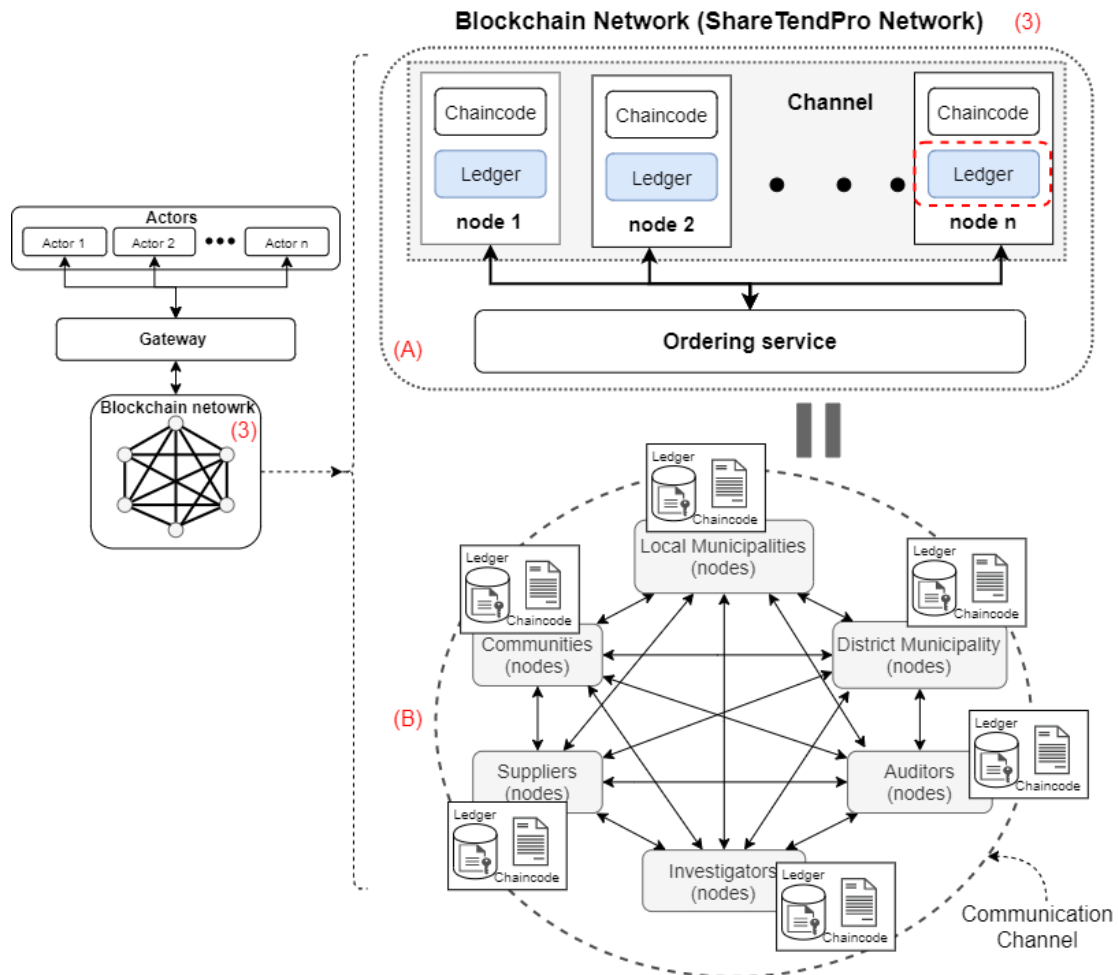


Figure 4.6 Blockchain network

The following section integrated all these components (actors, gateway, and Blockchain network) to generate the final step labelled number 4 as shown in Figure 4.2.

4.4.4 The ShareTendPro model as an integrated whole

This section integrates the components discussed in Figure 4.2 to generate the ShareTendPro model as our last step and these components are actors, gateway, and Blockchain network. As presented in the previous section, the actors' component aimed at identifying various actors that interact with tendering project information, and these actors are Local Municipalities, Communities, Suppliers, District Municipalities, Investigators, and Auditors as shown in

Figure 4.3. The gateway component seeks to manage the identities of these actors, including providing access to the Blockchain network as they interact with the Blockchain data or tendering project information. The Blockchain network component seeks to organise and distribute project information among various nodes or actors.

Figure 4.7 depicts a graphical representation of the ShareTendPro model as an integral of these components. It also reflects the flow of the project information as it passes through various components and objects. The numbers labelled 1 to 3 represent the three respective components, while number 4 can be viewed as the approach used by this study to explore how the proposed model integrates. The ShareTendPro model allows various actors to share tendering project information securely and efficiently. The Blockchain network component is one of the main key components that allow the ShareTendPro model to achieve its objectives because it is responsible for storing and sharing project information securely and efficiently. The flow of project information within the Blockchain network component is explored in detail in Figure 4.7.

In Figure 4.7, the Blockchain network component accepts all the transactions submitted by authorised actors, these transactions are then encrypted as part of preserving data integrity within the network. It will then take all these transactions and assess them based on the predefined requirements associated with each transaction as stipulated on the chaincode and if they meet all the requirements then they are declared as accepted transactions. As indicated in the previous section, all the accepted transactions are forwarded to the ordering service to be grouped into blocks. Thereafter, all the ordered transactions will be shared or distributed with all the nodes that are found within that network. Each node will then use the chaincode to verify these ordered transactions (blocks) before they can be appended or added to the ledger. This ledger consists of two elements namely: Blockchain data and world-state as shown in Figure 3.3 or Figure 4.7. The information stored in the Blockchain component is immutable by default, while the information stored in the world-state component of the ledger changes frequently as the state of the ledger changes. Hence, the Blockchain data can be viewed as the audit log or evidence storage since this information cannot be changed once it has been stored or recorded. Thereafter, all the actors that have a vast interest in a specific tendering project will now have access to the project information related to that project once it has been appended successfully.

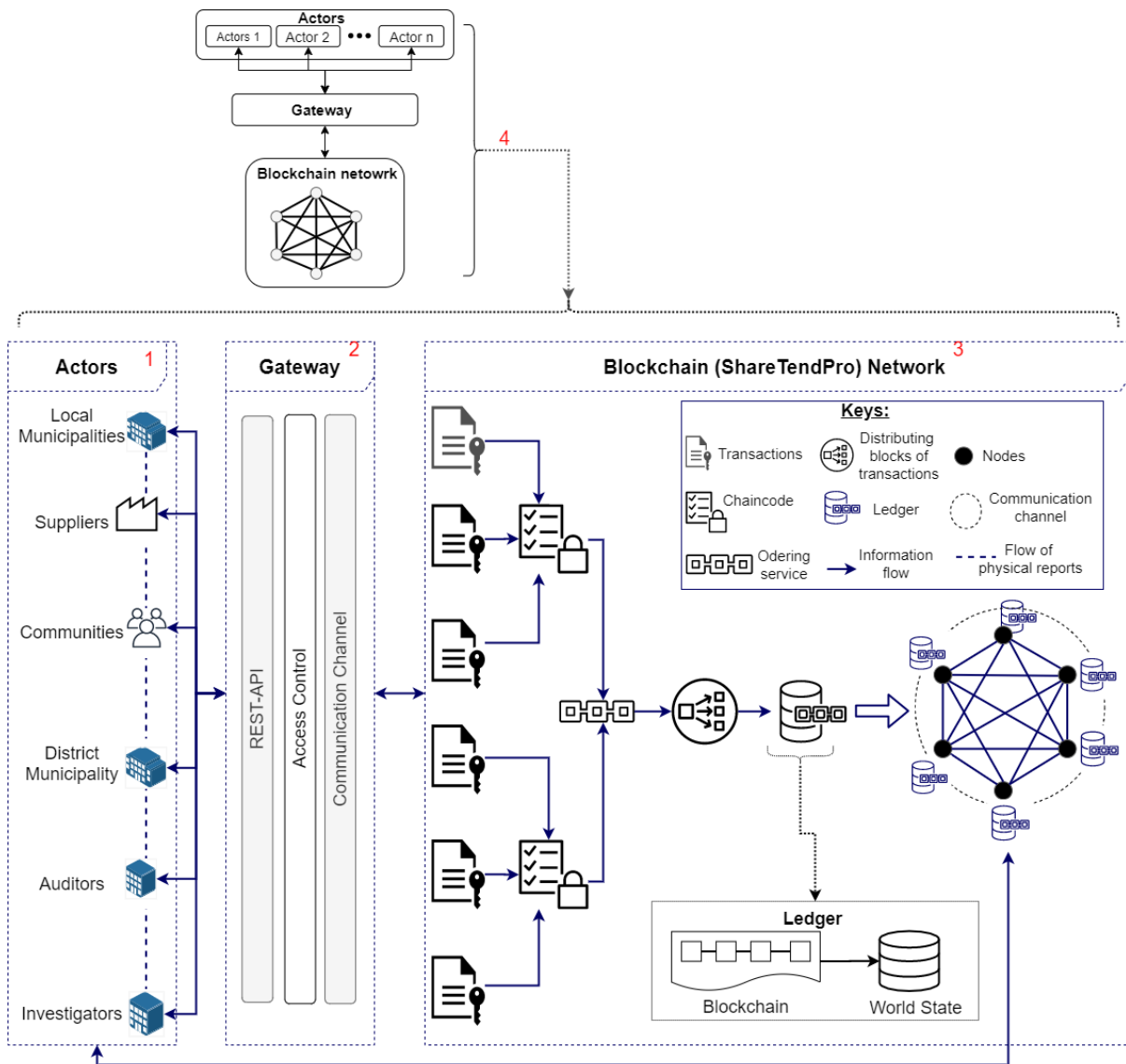


Figure 4.7 ShareTendPro model

Scenario 1 is used to demonstrate how the ShareTendPro model manages issues related to information security, transparency, and accountability within the tendering project. For instance, if Supplier S and a member of Municipality A agreed to falsify a tender project, whereby that member submitted a transaction to the ShareTendPro model that reflects the issuing of that tender project. The information related to that project and the details of the person who submitted the transaction will be stored permanently on the ledger, including the details of Supplier S. In addition, the project information will also be distributed to all the nodes within that network. Hence, the investigators or forensic experts will have access to this data wherever they are and conduct their investigation since the forensic data related to that tendering project is already stored on the ledger.

4.5 Conclusion

This chapter proposed a ShareTendPro model with an attempt to improve the current tendering system used by the SALG. The ShareTendPro model is based on the notion of using BCT to distribute project information to all the actors that have a vast interest in the tendering project. Additionally, this model seeks to promote information security, accountability, and transparency, including the audit trail as part of preserving digital evidence when it comes to fraud and corruption-related activities.

The following chapter details information concerning the design of the ShareTendPro model. Additionally, this chapter provides an in-depth view of the procedure to convert the proposed model (which is the ShareTendPro model) into a HLF network, including the scenarios that seek to explore how the proposed model work or represent its information as proof of concept.

5. Chapter 5: ShareTendPro model design

5.1 Introduction

The previous chapter detailed the ShareTendPro model that might be used by the South African Local Government (SALG) to securely distribute tendering project information with all the parties that have an interest in the project. This chapter presents a detailed design of the proposed ShareTendPro model, including its requirements specifications. In addition, this study makes use of Unified Modelling Language (UML) to visualise the design of the ShareTendPro model, including demonstrating how actors, objects, and the environment interact with each other.

The remaining details contained within this chapter is structured as follows; the requirement specifications of the ShareTendPro model are detailed, which includes the functional and non-functional requirements. The model design is discussed in detail in the following section. The system architecture of the ShareTendPro model is explored later. Thereafter, the last section provides a conclusion by summarising the entire chapter.

5.2 Requirements Specifications

Requirement specifications are one of the processes applied when designing and developing a software system or solution since it seeks to incorporate the constraints, behaviour, and services of the proposed solution. The success of a software solution can be determined by the extent to which various conflicting aspects and stakeholders' needs are managed with an intention to achieve the desired objectives. This study classifies these requirements specifications into two categories namely functional and non-functional requirements. The following sections explore these requirements in detail.

5.2.1 Functional requirements

The functional requirements detail what the software solution needs to accomplish by describing the interactions between the system and its environment using functions. In other words, the functional requirements focus on the functionality of the proposed solution or system. Therefore, the following items represent the functional requirements that should be achieved by the ShareTendPro model:

- a) *Adding resources* – it allows actors to add various resources into the Blockchain network based on the role they play. Some of these resources include the organisations and their members, contracts, tendering projects, and project information or reports.
- b) *Create Contract* – it allows Local and District municipality members to add information related to that contract that has been created, including assigning the relevant Supplier who will be responsible for providing such services.
- c) *Submit project reports or information* – it allows all actors (which are Investigators, Auditors, Suppliers, Communities, Local and District municipalities) to create and share project reports related to a specific tendering project.
- d) *Send events or notifications* – it allows the proposed solution to generate events as part of notifying relevant actors that a particular project report or information has been added to the network.
- e) *View project reports or information* – it should allow all the actors who have an interest in a specific tendering project to view all the reports related to that project.

5.2.2 Non-functional requirements

The non-functional requirements explore the constraints and behaviour that are not required by the actors, but there are required by the model to achieve certain objectives or tasks. However, the functionality of these non-functional requirements cannot be mapped out directly since they describe the model's properties or attributes. The following items depict the non-functional requirements of the ShareTendPro model:

- a) *Security* – since the project information will be shared among various actors, hence, intensive measures should be taken to ensure that all the information within the ShareTendPro model is well secured and protected from both external and internal threats. Therefore, the adoption of a technology that promotes information security while incorporating the use of encryption and cryptography is mandatory.
- b) *Scalability* – the model should support a large volume of data or project information without its performance being affected.
- c) *Availability* – all the actors that have an interest in the tendering project must be able to access it. Hence, the availability aspect of such a tool is crucial since it is aimed at enhancing the current tendering system used by the SALG. Additionally, the digital forensic investigator should be able to navigate through the entire history of the

tendering projects effortlessly since the proposed solution has a built-in audit trail feature.

- d) *Efficiency* – the efficiency of the model lies in the response time and data storage. In other words, the efficiency covers issues related to how fast it takes the proposed solution to perform certain tasks. For instance, all the project information will be distributed among all the actors that have an interest in that tendering project, unlike the current system whereby such information is shared with specific actors only, while other actors are required to submit proposal requests to access it.
- e) *Usability* – the model should be implemented in such a way that it is easy to use and operate. Hence, it should be user-friendly in such a way that all the actors should be able to achieve their desirable objectives or perform their tasks effectively and efficiently.
- f) *Reliability* – the model should provide consistent and accurate performances based on the intended functions. All the technical errors should be minimised and assigned a meaningful error message as part of simplifying the proposed solution, including eliminating its complexity.
- g) *Accessibility* – the ShareTendPro model will have web-based elements since it seeks to share or distribute project information among various actors located in different locations as part of accommodating the geographical aspect of it. Additionally, all the processes within the ShareTendPro model will be transparent in such a way that all the forensic data gathered using it could be used as evidence in a court of law.

As indicated earlier, all these requirements play an important role in designing and implementing the ShareTendPro model since it provides clear objectives required to address the problem identified in Chapter 1. The following section seeks to conceptualise the design of the ShareTendPro model.

5.3 Model design

This section presents a detailed design of the ShareTendPro model. However, this study classified the model design into three high-level categories namely: the architectural, behavioural, and structural design as shown in Figure 5.1. The architectural design represents the backbone or the architecture of the ShareTendPro model. The behavioural designs discuss the following UML diagrams: *use-case*, *state*, and *information flow* – and these diagrams focus on what must happen in the ShareTendPro model. In other words, the behavioural design is

used to describe the functionality of the ShareTendPro model. The structural designs explore the following UML diagrams: *class* and *package* – and these diagrams focus on the items or components that must be implemented in the ShareTendPro model.

Therefore, Figure 5.1 depicts a hierarchy that represents all the adopted designs as an overview of the model design.

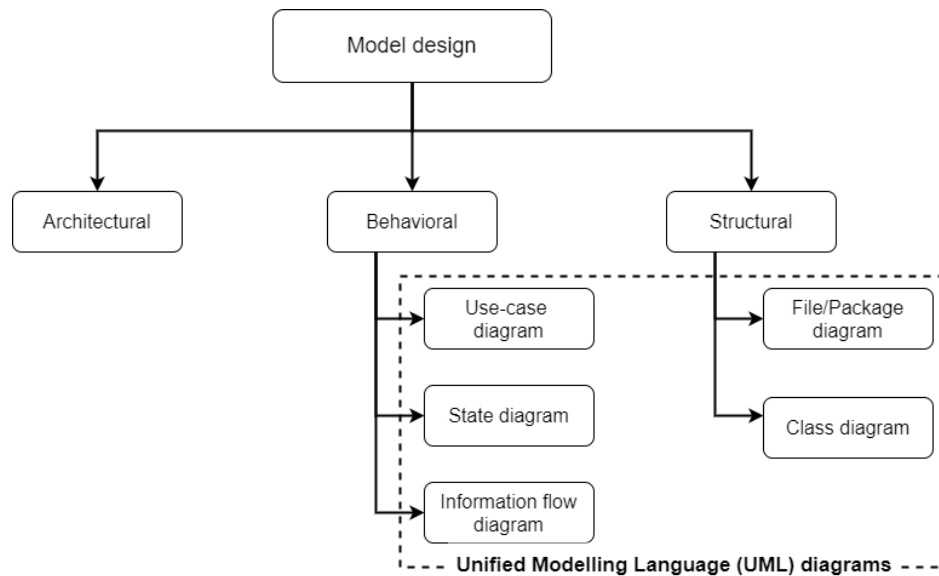


Figure 5.1 Model design strategy

The following section discusses the architectural design of the ShareTendPro model.

5.3.1 Architectural design (Model architecture)

This section discusses the model architecture of the proposed solution, which represents the conceptual architectural, behavioural, and structural design of the ShareTendPro model. This section makes use of a three-tier architectural design representing the logic of the ShareTendPro model. The three-tier architectural design consists of the following layers namely: *interface*, *business*, and *data* layers (as shown in Figure 5.2). These three layers are explored in detail below:

- *Interface layer (also known as the application layer)* – this layer focuses on the visual interaction by various actors to interact or interface with the application (ShareTendPro model). In other words, it focuses on the mechanism that enables actors to communicate or interact with the next layer (i.e., the business layer) to achieve their desired objectives. In this case, the interface layer depicts the REST-API since it allows actors to interact with the deployed network.

- *Business layer* – this layer focuses on the controller or mechanism used to facilitate the proposed solution. HLF (discussed in Chapter 3) is the controller mechanism used by this model to implement the proposed solution. However, HLF consists of many agents, and it uses these agents to perform or accomplish certain functionalities or tasks. Some of these agents used by HLF include Docker, Docker-compose, NodeJS, Go (also known as GoLang), JavaScript, and Python. The next chapter explores this in detail.
- *Data layer (also known as the network layer)* – this layer focuses on the mechanisms or processes used by the proposed solution to organise and store data and information. The proposed solution makes use of the ledger (particularly the distributed ledger system as discussed in detail in Chapter 3) to organise and store information. However, this process is governed by a chaincode. A chaincode can be viewed as a mechanism used by HLF to manage access and modification of the data within the network or ledger as discussed in Chapter 3.

Figure 5.2 represents the architecture of the ShareTendPro model. The numbering from 1 – 6 represents the logical flow of the information as it passes through these various layers (interface, business, and data layer). Additionally, these layers make use of the HTTP methods to communicate with each other and these HTTP methods contain one of the following operations: POST, GET, PUT, or DELETE. The POST operation is used to store data or information to the ledger. The GET operation is used to retrieve data or information from the ledger. The PUT operation is used to update the data or information, while the DELETE operation is used to delete certain information or data from the world state within the ledger or Blockchain network. However, the information stored within the Blockchain (a component within the ledger) cannot be deleted since this component act as an audit trail that records all the information or activities that seek to trigger the Blockchain network as discussed in Chapter 3. Figure 5.2 is explained in detail below.

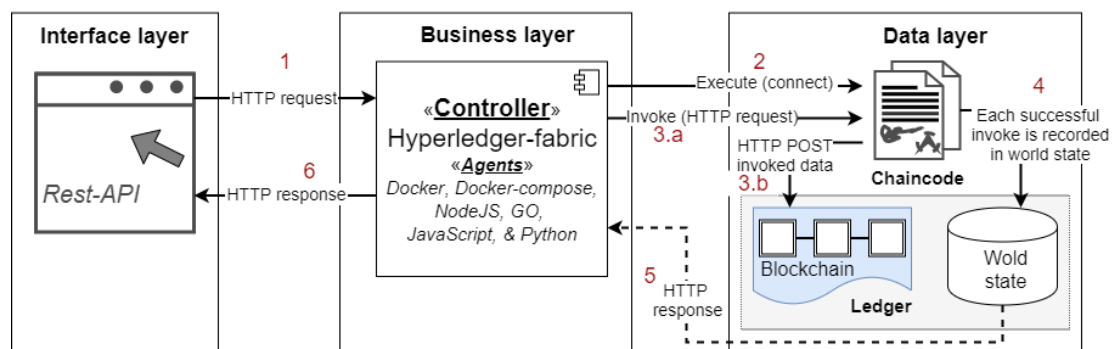


Figure 5.2 ShareTendPro model architecture

The following items explore the details in which these sequences of logical flow occur within the ShareTendPro model architecture (which is Figure 5.2):

- 1 – the REST-API submits an HTTP request that contains one of the following operations: POST, GET, PUT, or DELETE.
- 2 – the HLF makes use of the above request to execute the chaincode. This step can also be viewed as establishing the connection between the REST-API and the deployed network.
- 3 – Once the connection has been established, then HLF invokes the chaincode using the HTTP request data submitted in step 1 (represented by 3.a). Therefore, the information used to invoke chaincode is recorded to the Blockchain (which can be viewed as the transaction log) using the HTTP POST method (represented by 3.b).
- 4 – the actual data used to invoke the chaincode successfully is stored in the world-state, only if the operation associated with the request is either a POST or a PUT operation. Additionally, the other operations (GET and DELETE) only seek to either get or delete certain data within the world-state.
- 5 – Once an operation has interacted with the world-state, then an HTTP response is generated in relation to that specific request and forwarded to the controller (which is HLF).
- 6 – the above HTTP response is going to be passed to the REST-API that has submitted the HTTP request as part of signifying the completion of the process.

The following section discusses the behavioural design of the proposed solution, which is an in-depth discussion of how information flows within the ShareTendPro model since this section only explored the higher level of the logical information flow.

5.3.2 Behavioural design

This section discusses the behaviour or functionality of the ShareTendPro model. In other words, it explores how actors, objects, and environments interact with each other. The researcher acknowledges that they are various UML diagrams that might be used to explore the behavioural design of the ShareTendPro model. However, this study makes use of the following UML diagrams: use-case, state, and information flow (also known as the sequence diagram). The following sections explore these UML diagrams as applied within the ShareTendPro model.

5.3.2.1 Use-case diagram

A use-case diagram can be viewed as a scenario-based technique used to describe all the possible actors and their interactions with the ShareTendPro model. This research study has identified the following actors: District Municipalities, Local Municipalities, Communities, Suppliers, Auditors, and Investigators as presented in the previous chapter or shown in Figure 4.3. However, the actors represented in Figure 5.3 are reflected at a finer-grained level, i.e., where the actors in Figure 4.3 were mainly represented on an organisational level, the actors in Figure 5.2 are represented on the level of employees (members) working at those organisations. In addition, the Admin actor might fall under various categories hence it is depicted in a general form as shown in Figure 4.3. Therefore, the Admin actors are responsible for adding resources. These resources can include organisations and their members. The Local and District Municipality members have three main tasks assigned to them; the first task is to create a contract and assign a specific Supplier. The second task is to create a tender project and assign it to a specific contract, and the last task is to submit project reports related to a specific tender project. Additionally, the Local and District Municipality members are allowed to view project reports of all the tendering projects of their interest. The other members consist of the following actors: Suppliers, Communities, Investigators, and Auditors. These members are all responsible for submitting project reports of all the tender projects that fall within their mandate, and these members can also view other reports of the tendering projects of their interest.

Figure 5.3 represents the uses-cases of all the actors within the ShareTendPro model.

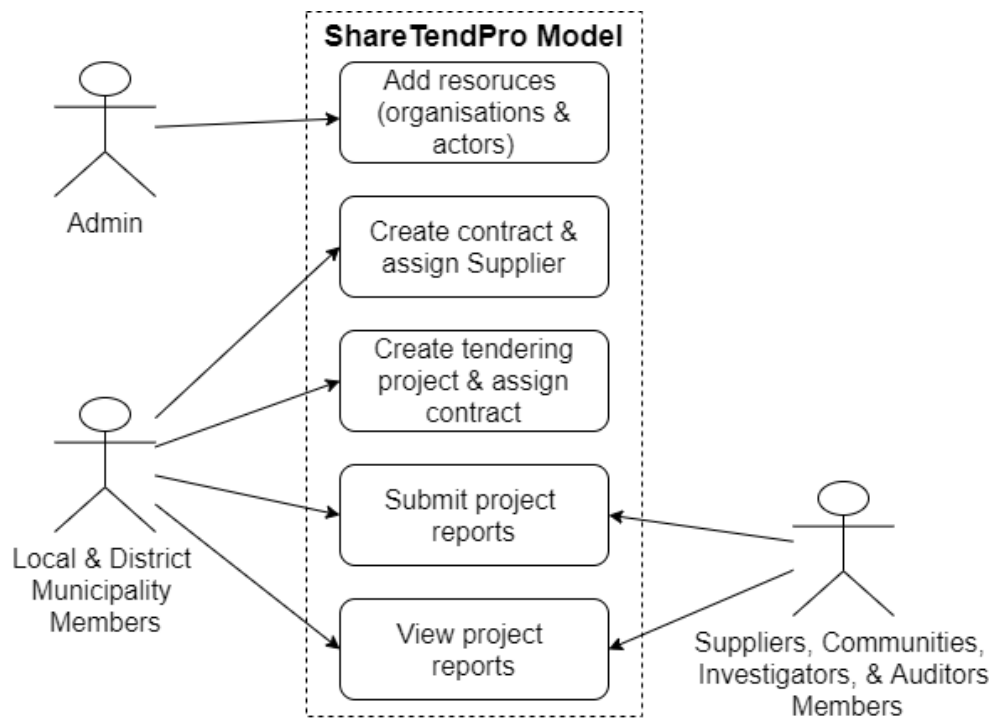


Figure 5.3 Use-case diagram

Figure 5.3 above depicts the use-case events associated with specific actors. The following section explores the sequence in which these events take place using a state diagram.

5.3.2.2 State diagram

A state diagram reflects how transactions take place between different service requests and these requests are the ones that trigger the state transition to move from one state (or transaction) to the next. As indicated in Chapter 3, the information stored within the world-state changes frequently as the network changes from one state to the other, as shown in Figure 5.4. This section explores the transactions that seek to trigger the state transitions within the ShareTendPro model.

Figure 5.4 below depicts a sequence of events within the ShareTendPro model that seek to trigger the world-state transitions as it changes from one state to the next. The numbering from T1 – T6 represents the transactions within the ShareTendPro model, while label A represents the option (a decision) used to verify certain information within the ShareTendPro model. The letter "T" in the numbering from T1 – T6 stands for "transaction", which implies that these objects represent various transactions that seek to trigger the world state of the proposed solution. In addition, these transactions can be mapped with the use-case events highlighted in Figure 5.4 using the roles associated with these actors as shown in Figure 5.4. For instance, T1

and T2 can be associated with adding resources in Figure 5.4, which reflect the admin roles. The following items explore the sequence in which these events occur:

- T1 – represents a transaction for adding organisations to the ShareTendPro model.
- T2 – represents a transaction for adding members to the ShareTendPro model and assigning them to their respective organisations.
- A – depicts an option in a form of a condition used to open or activate the functionalities of transactions 3 and 4, which can only be activated by either Local or District Municipality’s members or employees.
- T3 – represents a transaction for creating a contract and assigning that particular contract to a specific Supplier who will be responsible for executing the tendering project.
- T4 – represents a transaction for creating a tender project and assigning it to a specific contract created using transaction 3.
- T5 – represents a transaction for submitting project information or reports to the ShareTendPro model.
- T6 – represents a transaction for viewing project information or reports.

Therefore, Figure 5.4 represents the state diagram of the proposed solution. Figure 5.4 explored a high-level information flow of the ShareTendPro model. The following section explores the information flow of the ShareTendPro model in detail.

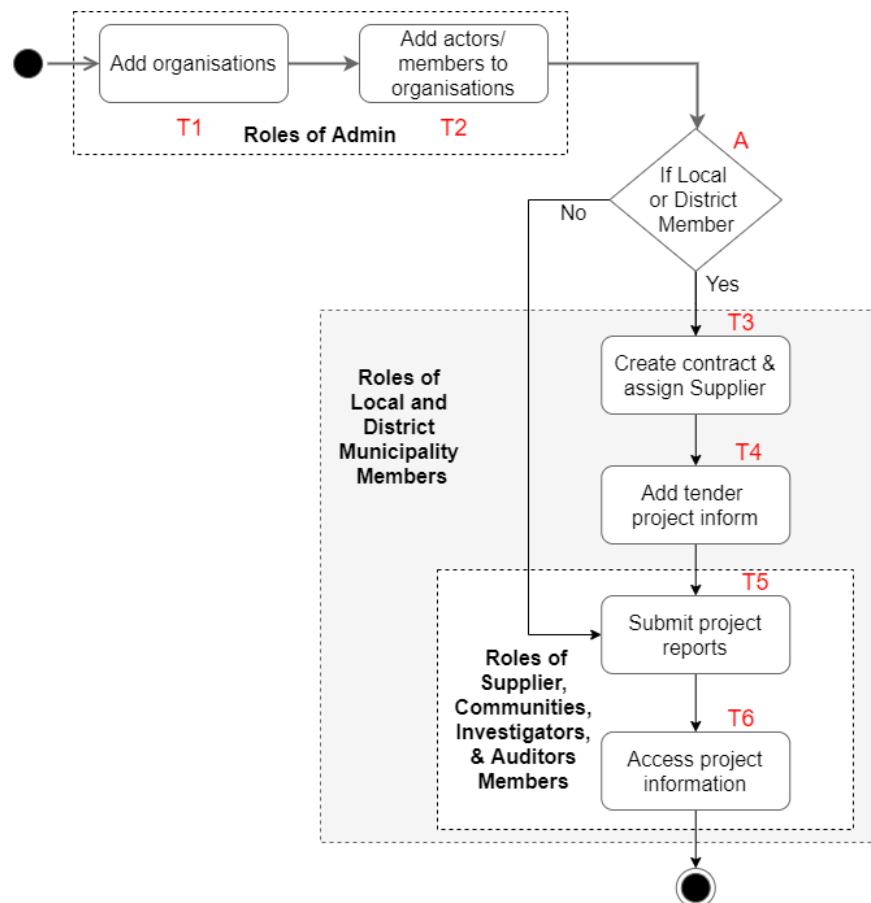


Figure 5.4 State diagram

5.3.2.3 Information flow

This section focuses on the information flow of the proposed solution, which explores the interaction between actors, objects, and the environment represented in a time sequence. In other words, this section details the chronological sequence of events required to accomplish a specific functionality of a given scenario. The researcher has divided this section into two sub-sections for the convenience of the reader to fully understand the chronological sequence of these events within the proposed solution. The first section explores the high-level of the information flow, while the second section explores the detailed information flow of the proposed solution.

5.3.2.3.1 Overview of the information flow

The details contained within this section explores a high-level of how the proposed solution works in general, which is the theoretical view or concept that seeks to explore the distributed nature of the proposed solution. Additionally, the information flow discussed in this section explores the interaction between some of the embedded objects and the surrounding

environment within the proposed solution. As illustrated in the previous chapter, this study adopts the use of HLF as the chosen Blockchain framework used to implement the proposed solution. Hence, HLF makes use of the ledger (distributed ledger system) to share project information with all the actors that have an interest in that tendering project. However, HLF consists of various nodes that seek to ensure that the network achieves the desired objective of sharing or distributing project information to all the actors securely and efficiently. These nodes can be classified into three categories namely: *clients*, *peers*, and *orderers* [96]. Therefore, the following items explore these nodes in detail:

- *Client nodes* – these are normal nodes or computers used by various actors to interact with the deployed network. Additionally, these nodes run an application that makes use of the REST-API to interact with the deployed network. Hence, all the actors are required to submit their transactions or project reports using this node (client node).
- *Peer nodes* – these nodes contain the chaincode (also known as smart-contract) and the ledger. The chaincode is used to govern the network by processing the transactions submitted by various actors, while the ledger is used to store or record these transactions submitted by various actors.
- *Orderer nodes (also known as ordering nodes or ordering services)* – these nodes are responsible for collecting all the accepted transactions within the Blockchain network and grouping them into blocks. Once the grouping of these transactions is complete, then the ordered transactions (which are blocks) are distributed or shared with all the peer nodes within the network.

HLF makes use of the Membership Service Provider (MSP) to manage the identities of all these nodes (peers, orderers, and clients) within the network. An MSP can be viewed as a component or mechanism within the network that issues credentials to all the nodes and actors that would be part of the network [97]. However, the identities of all the client nodes depend on the actor's credentials since the REST-API uses these credentials to authenticate and authorise the transactions submitted by various actors.

Figure 5.5 depicts a theoretical representation of the proposed solution, including how the available resources from different organisations can be utilised to accommodate all the actors. Additionally, Figure 5.5 also seeks to illuminate issues related to lack of personal computers and internet connection that might be experienced by some of the actors such as members of the Communities and Suppliers. Hence, actors will be able to utilise the municipality's resources whenever they want to submit their project information or reports related to a

particular tendering project. Figure 5.5 also represents how the identified nodes (clients, peers, and orderers) interact with each other to accomplish certain objectives within the network. However, all these nodes are represented using Docker containers within this study as supported by the HLF framework. A Docker container is a component that can be used to package a code or an application with all its dependencies (i.e., libraries) and deployed as one package [98]. Figure 5.5 is explained in detail below.

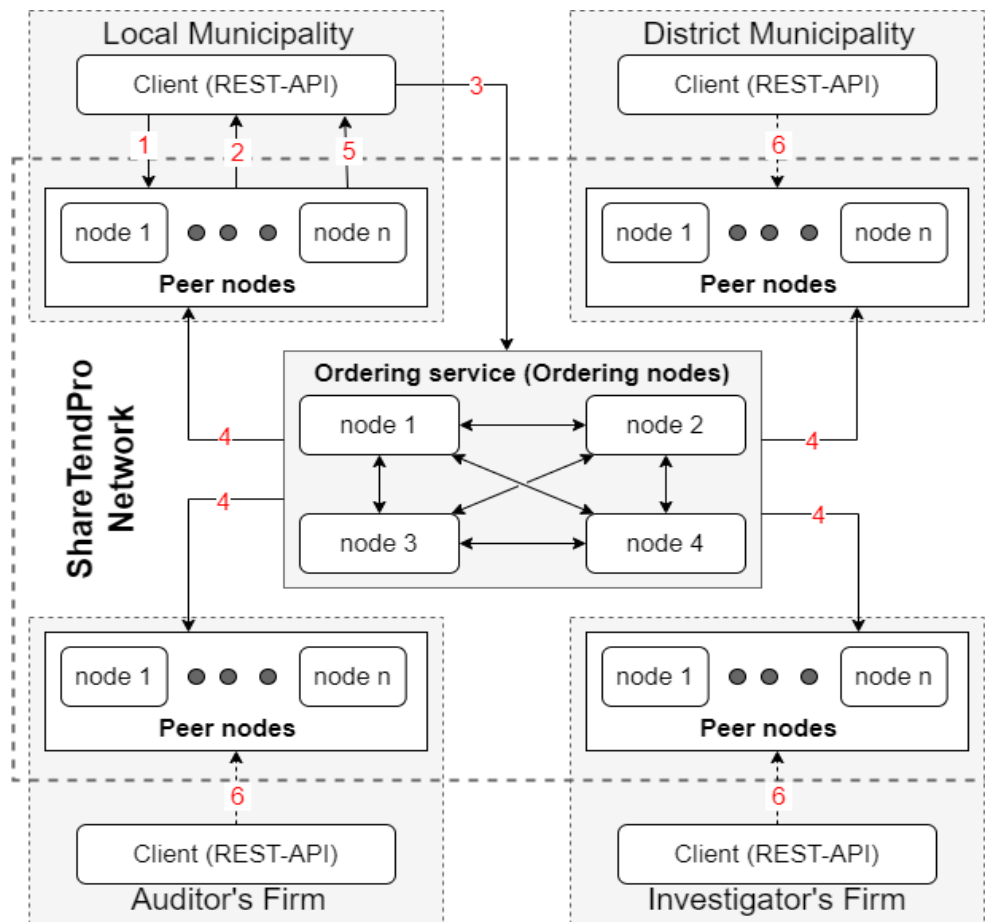


Figure 5.5 Overview of the information flow

Figure 5.5 consists of four organisations namely: Local Municipality, District Municipality, Auditors Firm, and Investigators Firm. Each of these organisations contains a client node (contains an application that runs the REST-API), and n number of peer nodes (which are labelled from node 1 to node n). Additionally, Figure 5.5 represents the four ordering nodes, and these nodes can be configured in such a way that they belong to either one or various organisations. The numbering from 1 – 6 represents the order in which these nodes interact with each other to achieve certain objectives within the network. Therefore, the following items explore this information flow in detail:

- 1 – the client node makes use of the REST-API to connect to the peer nodes within the network. Once the connection between the client and peer nodes is established, the network enables the client node to submit a transactional proposal to the network, requesting to either “query” or “update” the ledger.
- 2 – the peer nodes accept the proposal and examine it using chaincode as part of verifying whether it meets all the requirements associated with it. Thereafter, each peer node is going to endorse the proposal by attaching a digital signature and generate a proposal response for that request – thus, the “query” process is now complete. The endorsement process determines whether the transaction is accepted or rejected since it reflects the consensus reached by these various peer nodes within the network.
- 3 – the “update” process continues when the client node builds a transaction using the proposal responses submitted by various nodes and forwards it to the ordering service.
- 4 – the ordering service collects all these transactions across the network and groups them into blocks, which are then forwarded to all the peer nodes within the network for validation and committing. The validation process seeks to check the integrity of these transactions using chaincode, while the committing process seeks to append the transaction to the ledger.
- 5 – once all the peer nodes have updated the ledger, then an event is generated to notify the client node about the completion of the “update” process.
- 6 – signifies those other actors within the network will be able to view or access the updated ledger.

All these processes seek to ensure that the ledger within the peer nodes are kept up-to-date across the network. The following sections discuss how the information flow occurs between actors and objects for each respective transaction that seeks to trigger or interact with the ledger or Blockchain network. These sections represent various sequence diagrams as detailed information flow for each respective transaction within the proposed solution. As presented in the previous section, this study has identified the following members as actors of the proposed solution: Local Municipality, District Municipality, Supplier, Community, Auditor, and Investigator members. These actors can be subdivided into more fine-grained actors or roles (for example, Admin, Community, and Supplier are more fine-grained roles). However, the term 'actors' are consistently used in the remainder of the chapter. The transactions used within these sections, still correspond with the transactions identified in Figure 5.5. Each of these transactions consists of a number of events associated with it (typically labelled a to e). Note

that the initial transaction (T1 or detailed information flow of transaction 1) is explained in extensive detail. In later transactions, those details are omitted (simply by indicating that the details are the same as in a previous transaction).

5.3.2.3.2 Detailed information flow for transaction 1

This section discusses the information flow of transaction 1 (represented by *T1* in Figure 5.4 and Figure 5.6), which is one of the roles of the Admin actor. In other words, this section explores the sequents of events that occur when the Admin actor interacts with the deployed network using transaction 1, which focuses on adding the organisation details. Therefore, the following items explore the events associated with transaction 1 which requires the Admin actor to supply the details of the organisation that is interested in joining the ShareTendPro network as follows:

- a. The REST-API submits the interested organisation's details to the peer nodes in the form of a transactional request for adding data to the network.
- b. Recall that there could be n number of peer nodes involved in a voting scheme in order to verify a transaction as described in Chapter 3. Once the n peer nodes have reached a consensus, the transaction is accepted such that the chaincode verifies whether the transaction meets all the requirements associated with it (represented by *b1*). The chaincode, in this scenario, verifies whether all the necessary fields are filled before forwarding the endorsed transaction (by attaching digital signatures that reflect the voting scheme) to the REST-API (represented by *b4*). If some of the information is missing then an error message is generated and sent to the REST-API (represented by *b2*), however, the Admin actor will simply be prompted to complete missing data (represented by *b3*).
- c. The REST-API collects the results of the endorsed request and built a transaction, which will then be forwarded to the ordering service.
- d. The ordering service then orders these transactions by grouping them into blocks of the blockchain as shown in detail in Chapter 3. Thereafter, the ordered transactions are then forwarded to all the peer nodes within the network. These peer nodes validate the integrity of these ordered transactions using chaincode (represented by *d1*) before appending them to the ledger (represented by *d2*).

- e. After all the peer nodes have appended their ledger, an event is generated and sent to the REST-API (represented by *e1*) as part of the process to notify the Admin that the organisation details have been added successfully (represented by *e2*).

Figure 5.6 represents the sequence of events that explore the detailed information flow of transaction 1 as the Admin actor submits the details of the organisation that need to join the network.

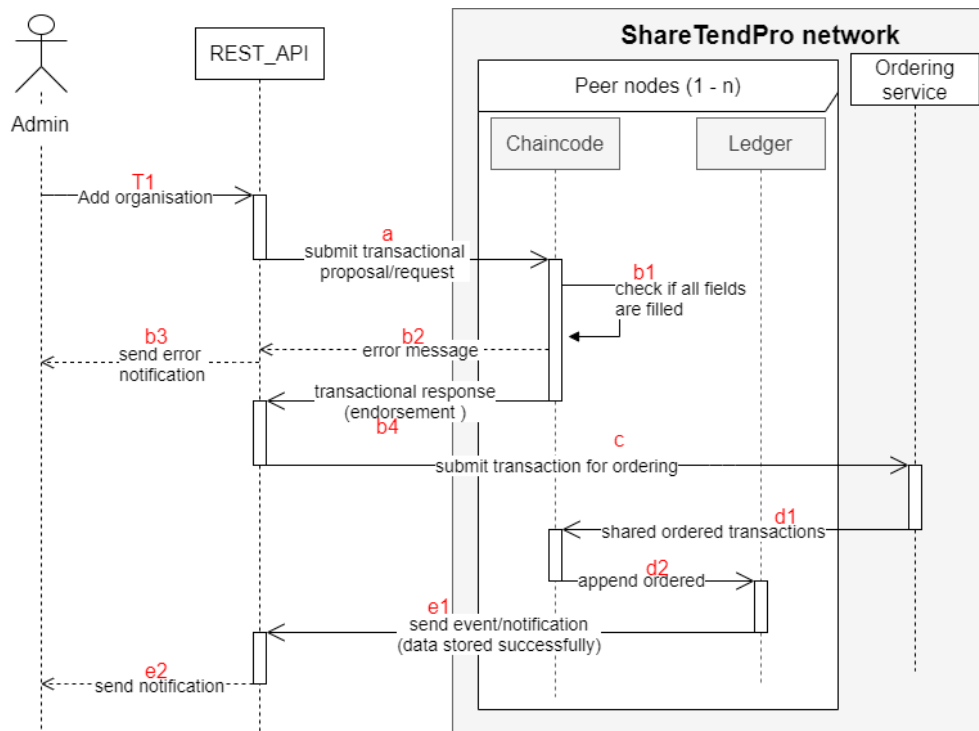


Figure 5.6 Details of transaction T1

5.3.2.3.3 Detailed information flow for transaction 2

This section explores the information flow of transaction 2 (represented by *T2* in Figure 5.4 and Figure 5.7) which is one of the roles of the Admin actor. This transaction *T2* allows the Admin actor to add members of the organisations to the ShareTendPro network as follows:

- a. The REST-API submits the members' details to the peer nodes or the blockchain network in a similar way as discussed in *T1.a*.
- b. This step is essentially the same as step *T1.b*, except for checking whether the assigned organisation to that particular member exists or not (represented by *b1* in Figure 5.7). If the assigned organisation details does not exist in the network, then an error message is generated (represented by *b2* in Figure 5.7) and sent to the REST_API, else the

chaincode will endorse the transaction before it can be forwarded to the REST-API (represented by *b4* in Figure 5.7).

- c. This step is the same as for *T1.c* when it comes to collecting the endorsed request built a transaction and forwarding them to the ordering service.
- d. This step is the same as for *T1.d* when it comes to forwarding ordered transactions to the peer nodes, using chaincode to validate the transactions, and appending these blocks of transactions to the ledger.
- e. This step is essentially the same as *T1.e*, except for notifying the Admin that the member details have been added successfully (which is represented by *e2* in Figure 5.7).

Figure 5.7 represents the sequence of events that explore the detailed information flow of transaction 2 as the Admin actor submits members' details to the ShareTendPro network.

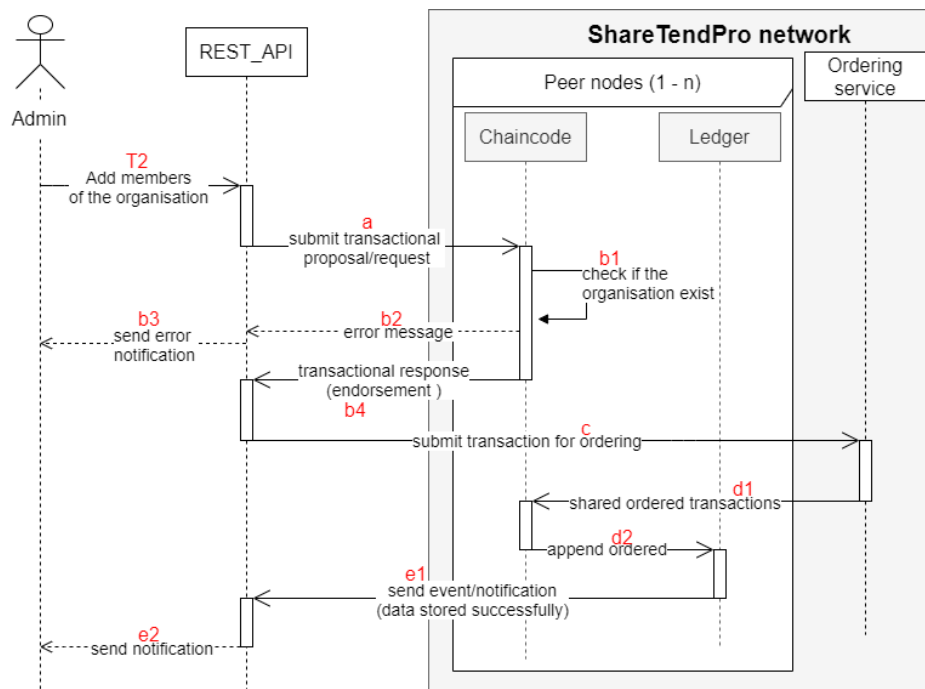


Figure 5.7 Details of transaction T2

The following section discusses the details of the information flow for transaction 3.

5.3.2.3.4 Detailed information flow for transaction 3

This section explores the information flow of transaction 3 (represented by *T3* in Figure 5.4 and Figure 5.8), which is one of the roles of either Local or District Municipality Member. This section focuses on the role played by the Local Municipality Member as part of exploring the detailed information flow for this transaction. Therefore, the process is the same for the District

Municipality Member, and will not be repeated. This transaction *T3* allows the Local Municipality member to create a contract and assign a specific Supplier who will be responsible for rendering that service as follows:

- The REST-API submits the contract and Suppliers' details to the peer nodes in a similar way as discussed in *T1.a*.
- This step is essentially the same as step *T1.b*, except for checking whether the member belongs to either a local or district municipality, including checking whether the Supplier assigned to that contract exists or not (represented by *b1* in Figure 5.8).
- This step is the same as for *T1.c* when it comes to collecting the endorsed request built a transaction and forwarding them to the ordering service.
- This step is the same as for *T1.d* when it comes to forwarding ordered transactions to the peer nodes, using chaincode to validate the transactions, and appending these blocks of transactions to the ledger.
- This step is essentially the same as *T1.e*, except for notifying the Local Municipality Member that a contract has been created successfully (represented by *e2* in Figure 5.8

Figure 5.8 depicts the sequence of events that explore the detailed information flow of transaction 3 as the Local Municipality member submits a transactional request for creating contract information.

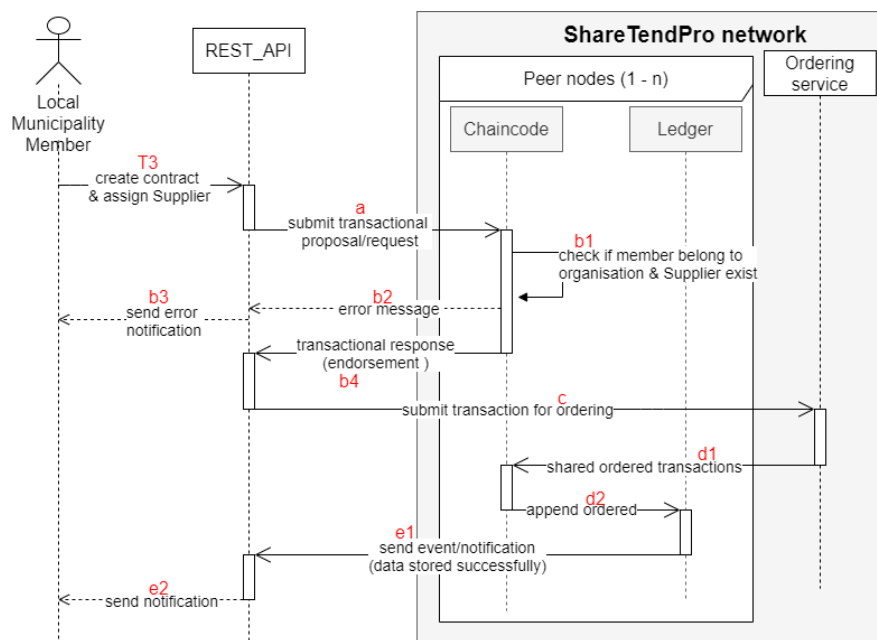


Figure 5.8 Details of transaction T3

The following section explores the details of transaction 4.

5.3.2.3.5 Detailed information flow for transaction 4

This section explores the information flow of transaction 4 (represented by *T4* in Figure 5.4 and Figure 5.9), which is one of the roles of either Local or District Municipality Member, except for the Admin actor. However, this section only focuses on the role played by the Local Municipality member as part of exploring the detailed information flow for this transaction. Therefore, this transaction *T4* allows the Local Municipality member to create a tendering project and assign it to a contract created in *T4* as follows:

- a. The REST-API submits the tendering project details to the peer nodes in a similar way as discussed in *T1.a*.
- b. This step is essentially the same as step *T1.b*, except for checking whether the contract has not been assigned any tendering project (represented by *b1* in Figure 5.9).
- c. This step is the same as for *T1.c* when it comes to collecting the endorsed request built a transaction and forwarding them to the ordering service.
- d. This step is the same as for *T1.d* when it comes to forwarding ordered transactions to the peer nodes, using chaincode to validate the transactions, and appending these blocks of transactions to the ledger.
- e. This step is essentially the same as *T1.e*, except for notifying an actor who submitted the transaction that a tendering project has been created successfully (represented by *e2* in Figure 5.9).

Figure 5.9 depicts the sequents of events that explore the detailed information flow of transaction 4 as the Local Municipality member submits a transactional request for creating tendering project information.

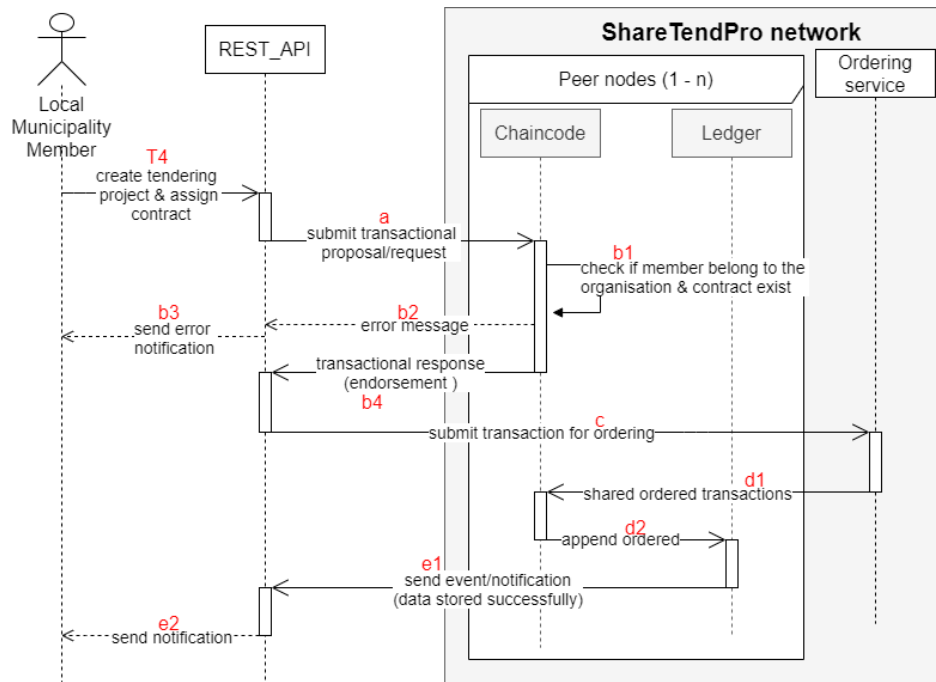


Figure 5.9 Details of transaction T4

The following section explores the details of transaction 5.

5.3.2.3.6 Detailed information flow for transaction 5

This section explores the information flow of transaction 5 (represented by *T5* in Figure 5.4 and Figure 5.10), which is one of the roles assigned to all the actors, except for the Admin actor. However, this section also focuses on the role played by a Supplier member as part of exploring the detailed information flow for this transaction. Therefore, this transaction *T5* allows all the actors to submit project information or reports as follows:

- The REST-API submits project information or reports details to the peer nodes in a similar way as discussed in *T1.a*.
- This step is essentially the same as step *T1.b*, except for checking whether that particular member is permitted to submit project information or report to that particular tendering project or not (represented by *b1* in Figure 5.10).
- This step is the same as for *T1.c* when it comes to collecting the endorsed request built a transaction and forwarding them to the ordering service.
- This step is the same as for *T1.d* when it comes to forwarding ordered transactions to the peer nodes, using chaincode to validate the transactions, and appending these blocks of transactions to the ledger.

- e. This step is essentially the same as *T1.e*, except for notifying an actor who submitted the transaction that the project information or report has been added successfully (which is represented by *e2* in Figure 5.10).

Figure 5.10 depicts the sequential events that explore the detailed information flow of transaction 5 as the Supplier member submits a transactional request for adding project information or reports to the network.

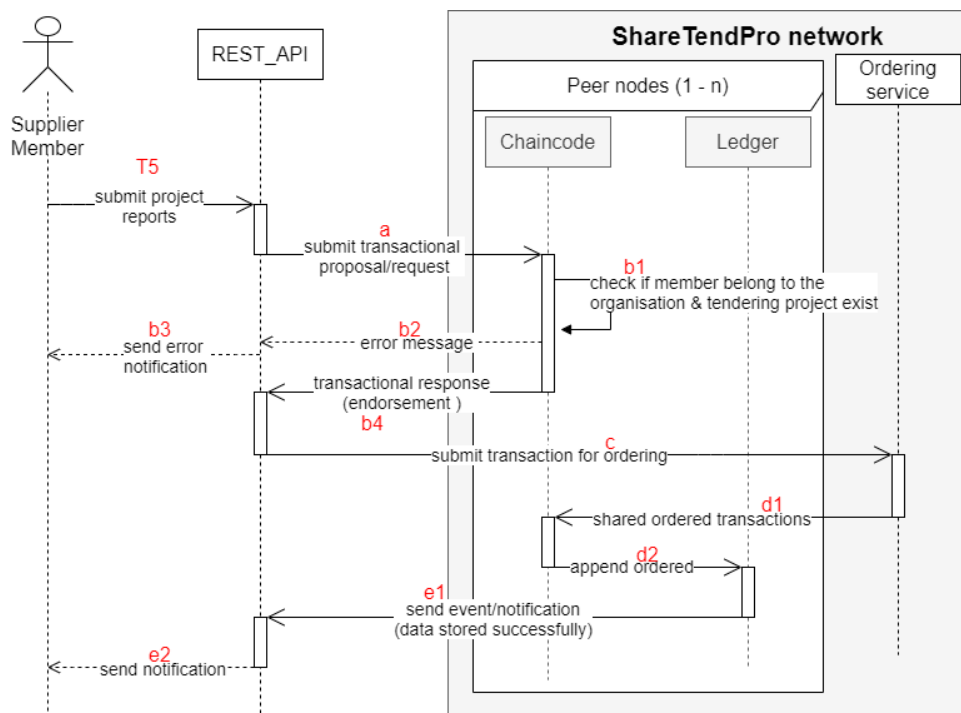


Figure 5.10 Details of transaction T5

The following section explores the details of transaction 6.

5.3.2.3.7 Detailed information flow for transaction 6

This section explores the information flow of transaction 6 (represented by *T6* in Figure 5.4 and Figure 5.11) which is one of the roles assigned to all the actors, except for the Admin actor. This transaction *T6* allows all the actors to view or access project information or reports as follows:

- The REST-API submits a transactional request to view or access certain tendering project information to the peer nodes in a similar way as discussed in *T1.a*.
- This step is essentially the same as step *T1.b*, except for checking whether that particular actor is permitted to access such project information, including checking whether the tendering project itself exists or not (represented by *b1* in Figure 5.11).

Once all these requirements are met then that particular request is converted into a query statement and forwarded to the ledger as part of a statement or condition to retrieve specific information of a particular tendering project (represented by *b4* in Figure 5.11).

- c. The ledger will then access the query statement and retrieve that specific information requested and send a notification to the REST-API (represented by *c1* in Figure 5.11). Thereafter, the REST-API will then notify that particular actor who requested the data that the project information or reports have been retrieved successfully (as shown in *c2* in Figure 5.11).

Figure 5.11 depicts the sequential events that explore the detailed information flow of transaction 6 as the Community member submits a transactional request for viewing or accessing certain project information or reports.

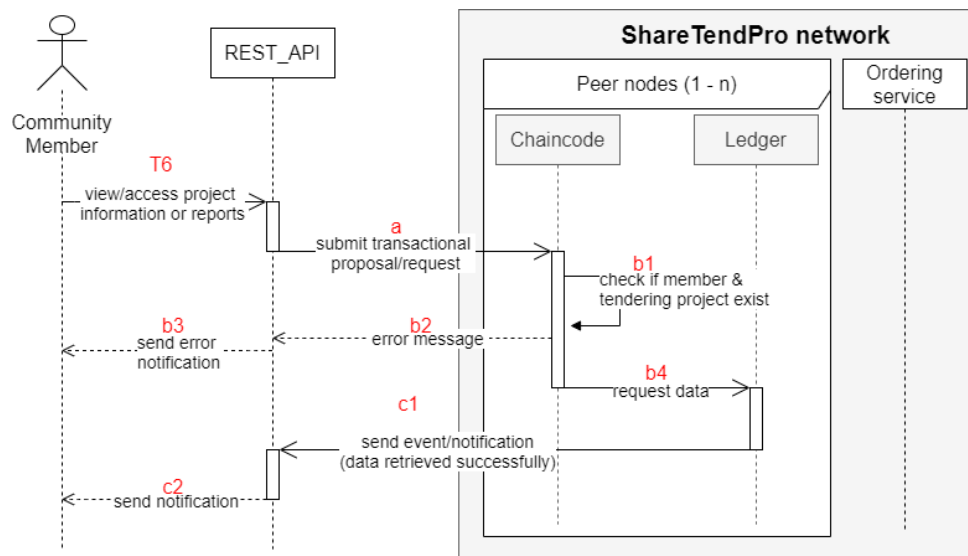


Figure 5.11 Details of transaction T6

The following section detailed the overall information flow, which is the integration of all these transactions *T1 – T6*.

5.3.2.3.8 Overall detailed information flow

This section seeks to integrate all the above-mentioned transactions from *T1 – T6*, including highlighting some of the transactions that were omitted (for example the actors of *T5* and *T6*).

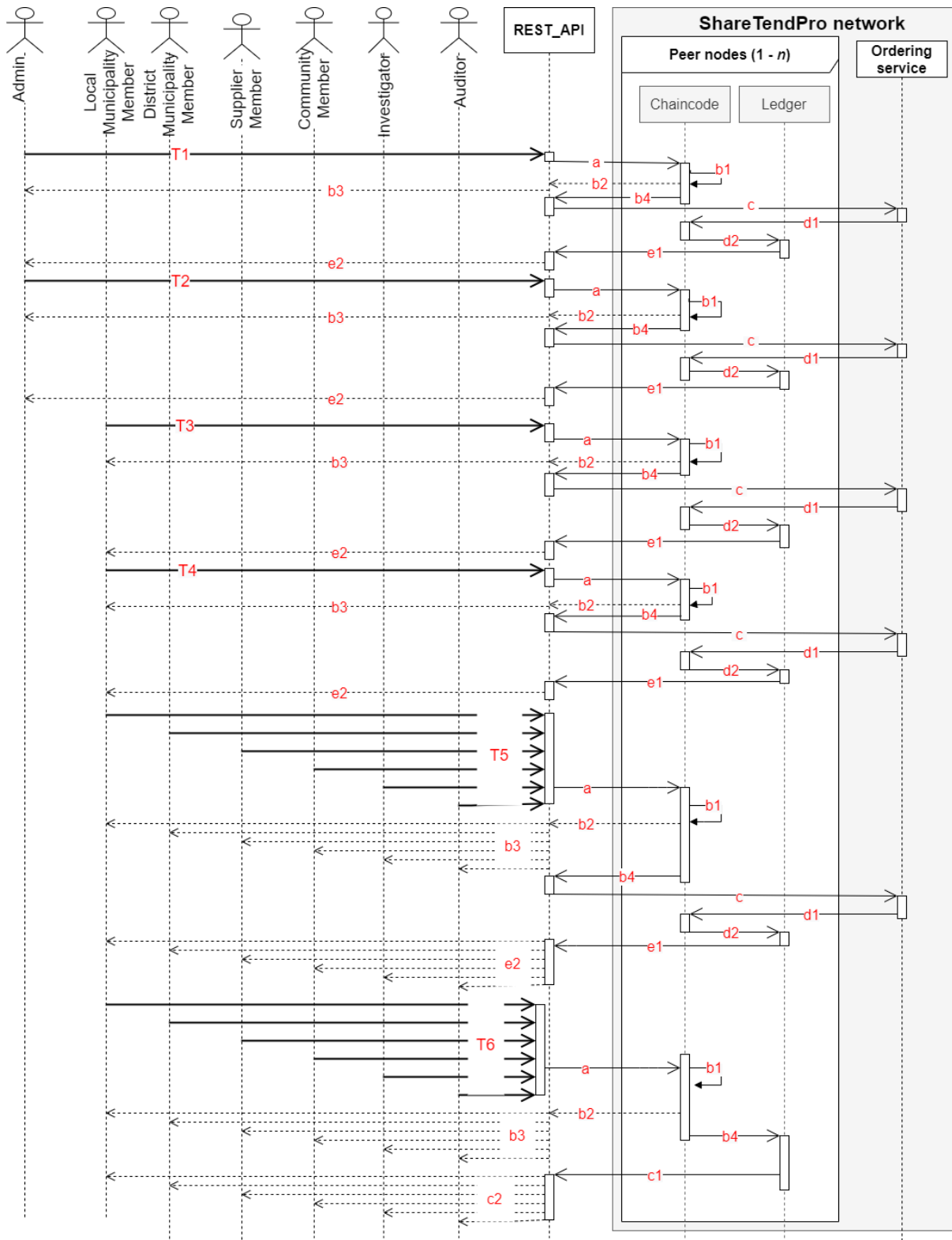


Figure 5.12 Detailed information flow

All these UML diagrams (use-case, state, and information flow) discussed above explored the behavioural design of the proposed solution. Therefore, the following section explores the diagrams that seek to discuss the structural design of the proposed solution.

5.3.3 Structural design

This section explores the structures or components that need to be implemented by the proposed solution. There are various diagrams that might be used to represent the structural design of the proposed solution. However, this study makes use of the following UML diagrams: class diagram and package diagram to represent the structural design of the ShareTendPro model. The following sections discuss these structural diagrams in detail, starting with the class diagram and followed by the package diagram.

5.3.3.1 Class diagram

The class diagram is one of the structural design diagrams that seeks to use entities, attributes, operations (or methods), and the relationships among entities to describe the data structure of the proposed solution. In other words, the class diagram can be viewed as a process to visualise the data model of the proposed solution. This study divides the class component within the class diagram into three sections namely: top, middle, and bottom section for the convenience of the reader to understand the data structure used by the proposed solution. The top section contains the name of the class or entity. The middle section contains the attributes of the class or entity, while the bottom section contains the operations or methods associated with that class or entity. For instance, the details of the Organisation class (labelled 1) in Figure 5.13 are as follows:

- The name of the class is “*Organisation*”,
- The attributes of this class are: “*OrgID*”, “*Name*”, and “*Address*”,
- The method or operation used by this class is “*AddOrganisation()*”.

This class had only one method, however, note that the other classes have their own different attributes and methods. Figure 5.13 represents the data model of the proposed solution using a class diagram. The numbering from 1 – 4 depicts the main four classes within the proposed solution since other classes are derived from one of these four classes. For instance, the classes labelled Org1 – Org6 are derived from class 1, while the classes labelled M1 – M6 are derived from class 2. Some of these classes depend on the data stored by other classes to achieve their desired functionalities or objectives. For example, class 2 depends on class 1 since class 2 requires the organisation details for it to successfully store the details of that member. Additionally, class 3 depends on the information stored in class 1 and 2, while class 4 depends on the information stored in class 3. However, some of these dependencies are not visible or

clear enough in Figure 5.13 since the diagram only portrays the information or data stored within that class.

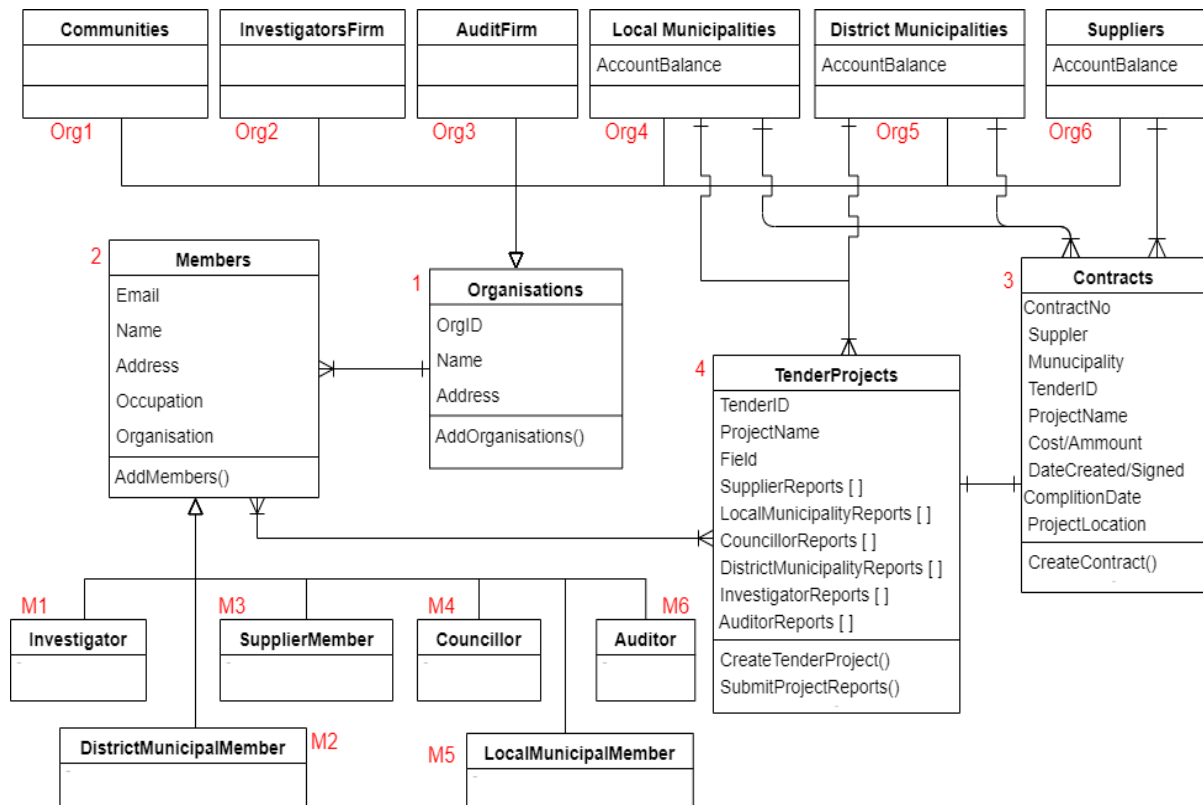


Figure 5.13 Class diagram

The numbering from Org1 – Org6 illustrates the relationship in which these classes have with class 1, whereby class 1 can be viewed as the parent of all these classes labelled Org1 – Org6. The same notion applies to the relationship between class 2 and the classes labelled M1 – M6. As highlighted in the previous section, only the Local and District Municipalities are allowed to create contracts and tendering projects. An association whereby one or many contracts and tendering projects belong to a municipality is also highlighted in Figure 5.13 (i.e., referred to as a one-to-many relationship). In addition, one or many contracts might also be awarded to a specific Supplier, that is also reflected in the diagram.

The following section discusses the package diagram, which focuses on the important files within HLF. These files are used to configure some of these classes highlighted in Figure 5.13. For instance, the crypto-config file (represented in Figure 5.14) contains a blueprint of the network topology since it depicts information related to the configurations of the organisations (class “Organisation” and its children Org1 to Org6), members of the organisations (class “Members” and its children M1 to M6), and the number of peer nodes that belong to a specific organisation.

5.3.3.2 Package diagram

This section explores some of the important packages incorporated together to generate the ShareTendPro model solution. Additionally, this section only focuses on the packages or files that should be considered during the configuration and deployment of the proposed solution. Hence, this study discusses the following three important files namely: *crypto-config*, *configtx*, and *docker-compose* file as used by the HLF network. Therefore, the following items explore how HLF incorporate these files together:

- *Crypto-config file (also known as cryptogen file)*: this file contains the network topology of the proposed solution. Additionally, it is also used to generate cryptographic materials that are used to uniquely identify all the authorised nodes and users within the network [99]. These crypto materials are nothing, but the certificates and keys used by the network to identify all the organisations and their components or resources, including users. The configuration details contained within these files are discussed in more detail in the next chapter.
- *Configtx file*: this file enables the network to generate the genesis block (which is the first block in the Blockchain) and distributes that block to all the ordering and peer nodes within the network. This file also allows the network to create channel artifacts, which enables certain peer nodes to communicate or share project information secretly. Additionally, this file contains the configurations of specific peer nodes that can be assigned the role of anchor peers, which enables them to be discovered by any nodes within the networks. A practical example of how anchor peers work is the way in which the news bulletins are aired or broadcasted since an anchor reporter or journalist is the one who sits in a studio and collects all the latest news feeds from other reporters or journalists in different locations and broadcasts them. The same notion applies to anchor peers since they receive all the transactions submitted by various actors and endorse them before they can be forwarded to the ordering service. The configuration of this file is explored in the following chapter.
- *Docker-compose file*: this file enables the network to collect all the necessary information generated by the above two files and create Docker containers that store information related to a particular component within the network. Additionally, it also allows the network to run that particular component as one package since all the dependencies and crypto materials of that particular component will be stored in a specific docker container. For instance, if a specific organisation within the network

consists of two peer nodes, then the network is going to generate two docker containers that will be used to store all the crypto materials associated with these peer nodes.

Figure 5.14 depicts these packages or files (*crypto-config*, *configtx*, and *docker-compose* file) used by HLF to generate the desired proposed solution.

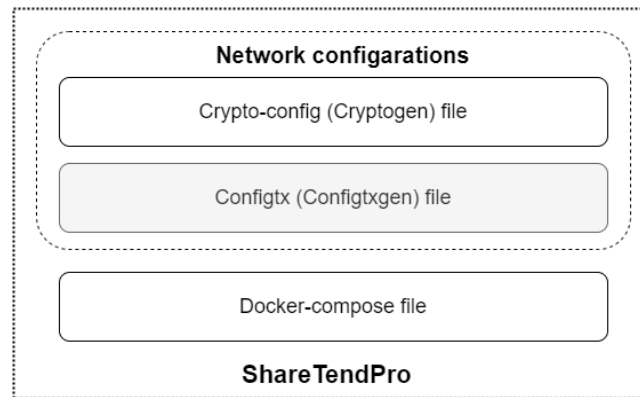


Figure 5.14 Package diagram

5.4 Conclusion

This chapter discussed the ShareTendPro model design, which includes the requirements specifications and model design. The requirements specifications explored the functional and non-functional requirements of the proposed model, while the model design focused on the architectural, behavioural, and structural design of the proposed model. The model design of the proposed solution, as shown throughout this chapter, consisted of several UML diagrams, and each of these diagrams depicted a specific event or component within the proposed solution.

The following chapter delves deeper into the ShareTendPro model by discussing the implementation of the model.

6. Chapter 6: Implementation of the ShareTendPro prototype

6.1 Introduction

Chapter 4 of this study proposed the ShareTendPro model that seeks to address the identified problem of using paperwork to share tendering project information, while Chapter 5 explored the detailed design of the proposed solution, which is called the ShareTendPro model. Chapter 5 further discussed the ShareTendPro model, and the components associated with it were fully explained.

This chapter is devoted to the implementation of the ShareTendPro model as a prototype. Note that a fully-fledged chapter with all the details of the implementation is provided in Appendix B. Hence, only the crux of the implementation is given in this chapter. Additionally, note that this entire chapter does not show the execution of the prototype (as purported by the model design of Chapter 5), but strictly its implementation details. Chapter 7 shows the running of the prototype in detail, as the results of the ShareTendPro model. Therefore, the remainder of this chapter is structured as follows: the set of tools used to implement the prototype are discussed in relative detail, which includes hardware and software specifications. The ShareTendPro model implementation is discussed in relative detail in the next section. Additionally, this section explores the implementation of the ShareTendPro network topology and the chaincode that governs the ShareTendPro network topology. The last section provides a conclusion by summarising the chapter.

6.2 Tools used to develop the prototype

This section mentions the tools used to implement the proposed solution, which is the ShareTendPro prototype. However, a fully detailed section is contained in Section B.2 of Appendix B. As indicated in Appendix B, this section is classified into two categories namely hardware and software requirements. The hardware requirements focus on the requirements defined by the operating system used by the proposed solution, while the software requirements focus on the software tools installed within the virtual machine. Additionally, the hardware requirements are classified into two categories namely physical and virtual machines. The physical machine category details the requirements related to the hardware-based device or computer used, while the virtual machine category details the requirements related to the virtual computer that seeks to emulate the actual physical machine.

All these requirements play a critical role because they form part of the pre-requisite for the proposed solution. However, it is mandatory to ensure that all the Software requirements (presented in Table B.2 of Appendix B) are met since they are aimed at ensuring that the chosen Blockchain framework, which is HLF, runs or works successfully. Therefore, the following section delves into the implementation of the ShareTendPro model to accomplish this.

6.3 ShareTendPro model implementation

This section represents an implementation of the ShareTendPro model. However, a detailed implementation section is contained in Section B.3 of Appendix B. As indicated in Figure 6.1, this section is classified into two namely ShareTendPro network topology and chaincode development. The ShareTendPro network topology focuses on the configuration details used to implement the virtual or Blockchain network of the proposed solution, while the chaincode development focuses on implementing the mechanisms or rules that govern the ShareTendPro network. Additionally, the ShareTendPro network topology category is classified into three categories namely: *crypto-config*, *configtx*, and *docker-compose* as shown in Figure 6.1. These categories explore all the configuration details needed for implementing the desired ShareTendPro network topology (which is explained in detail later). Figure 6.1 depicts the adopted hierarchy of the ShareTendPro development process. Note that the process order in which the model was developed, is shown by the numbers 1 to 4, as indicated by the key in the figure. A more detailed discussion of the ShareTendPro development process follows next.

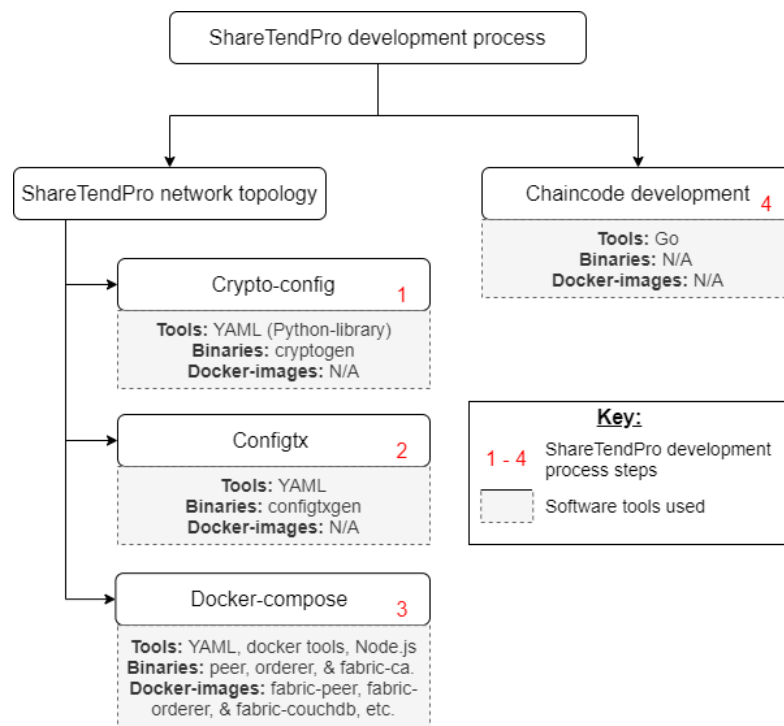


Figure 6.1 ShareTendPro development process high-level

As indicated in Figure 6.1, the configuration details related to the *crypto-config* category are compiled using the YAML library and fed to a binary file called *cryptogen* to generate the crypto materials that would be used to secure the communication within the proposed solution. The configuration details related to the *configtx* category are compiled using the YAML library and fed to the *configtxgen* binary file to generate channel artifacts. Channel artifacts can be viewed as the materials used by the Blockchain network to generate a private communication channel that allows various organisations to secretly share project information. The configuration details related to the *docker-compose* category are compiled using the YAML library and fed to the docker-compose tool. The docker-compose tool then uses the docker-engine tool to generate the virtual nodes required by the proposed solution. Thereafter, the Node.js tool would then use these virtual nodes to create a virtual network or a Blockchain network of the proposed solution. However, a specific virtual node requires certain binaries and docker images for it to be regarded as a successful virtual node of the ShareTendPro network, which is the Blockchain network used by the proposed solution.

The details of the chaincode are compiled using the Go tool, which contains functions that allows various nodes within the proposed solution to interact with the virtual network of the proposed solution. Hence, all the functions within the chaincode development can be viewed as the rules that govern the virtual network of the proposed solution.

The following sections details the implementation of the ShareTendPro model, starting with the implementation detail of the ShareTendPro network topology, followed by the implementation details of the chaincode development.

6.3.1 ShareTendPro network topology

This section explores all the necessary information required to generate the desired Blockchain network of the proposed solution. However, a fully detailed section is contained in section B.3.1 of Appendix B. As indicated in Figure 6.2, this section is divided into three categories namely: *crypto-config*, *configtx*, and *docker-compose*, and all these categories correspond with the three important files discussed in the previous chapter. However, the ShareTendPro network consists of other script files that might be used to either package some of the command lines required to run the network or generate the network artifacts. A script file can be viewed as a text document that contains certain instructions written in a specific scripting language, hence, these files are both human-readable and machine-readable. Therefore, Figure 6.2 depicts a more detailed hierarchy of the ShareTendPro network topology development process, compared to Figure 6.1. A more detailed discussion of the ShareTendPro development process follows next.

The following items detail the information related to the three categories (i.e. *crypto-config*, *configtx*, and *docker-compose*) highlighted in Figure 6.2 and how these categories correspond with the three important files (*crypto-config.yaml*, *configtx.yaml*, and *docker-compose.yaml*) discussed in the previous chapter:

- **Crypto-config:** contains the configuration details related to the *crypto-config.yaml* file. However, the *crypto-config* category is divided into three sections namely:
 - *ShareTendPro network design* section seeks to visualise the desirable Blockchain network of the proposed solution.
 - *Crypto-config.yaml file* section seeks to explore the configuration details of all the organisations and their resources (nodes and users).
 - *Crypto-config folder* section seeks to examine the cryptographic materials generated after executing the *crypto-config.yaml* file.

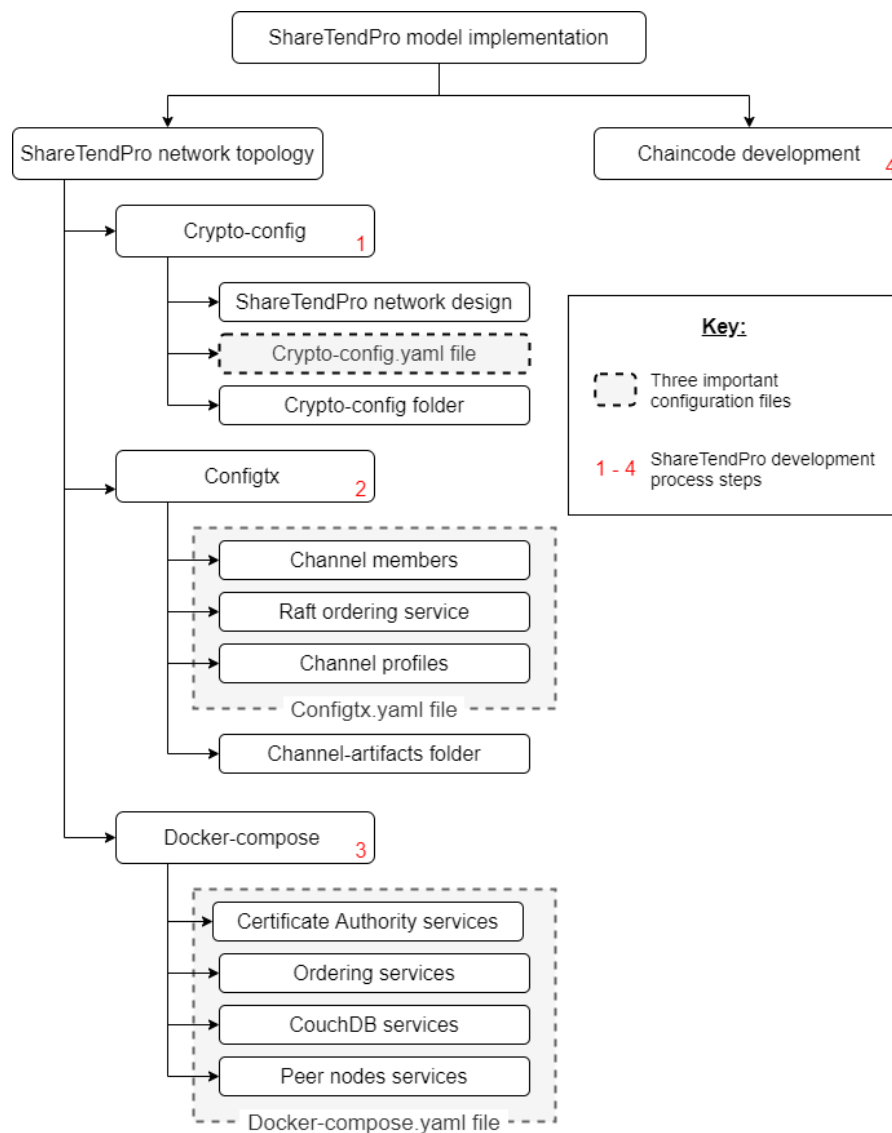


Figure 6.2 ShareTendPro network topology development process

- **Configtx:** depicts the configuration details related to the *configtx.yaml* file. The configtx category is divided into four sections namely:
 - *Channel members* section contains the configuration details of all the organisations that form part of the communication channel.
 - *Raft ordering service* section contains the configuration details of the ordering service used by the proposed solution.
 - *Channel profiles* section contains the configuration details related to the profiles that seek to generate the channel-artifacts.
 - *Channel-artifacts folder* section examines the results generated after executing the channel profiles within the *configtx.yaml* file and these results are stored within the channel-artifacts folder.

- **Docker-compose:** contains all the configuration details related to the *docker-compose.yaml* file. As indicated in Figure 6.2, the docker-compose category is classified into four sections namely:
 - *Certificate authority services* section explores all the configuration details related to all the services that seek to identify the resources that belong to a specific organisation within the Blockchain network.
 - *Ordering services* section depicts the configuration details related to all the services that seek to add new transactions to the Blockchain network.
 - *CouchDB services* section explores all the configuration details that seek to store the state of a Blockchain network.
 - *Peer node services* section explores all the configuration details of the peer nodes that belong to various organisations, such as Local Municipality, District Municipality, Investigators' Firm, and Auditor's Firm.

This section discussed the necessary information required to generate a network topology of the proposed solution, which is the ShareTendPro model. The following section focuses on the development of the rules (also known as chaincode) that seek to govern the Blockchain network. In other words, the following section explores the chaincode development that allows various organisations (i.e. Local Municipality, District Municipality, Investigator's Firm, and Auditor's Firm) to interact with the ShareTendPro network of the proposed solution.

6.3.2 Chaincode development

This section explores all the necessary information required to implement the chaincode that would be used by the proposed solution to govern the Blockchain network. However, a detailed implementation section is contained in section B.3.2 of Appendix B. As indicated in Figure 6.3, the chaincode development is divided into two categories namely *tender structure* and *invoke functions*. The *tender structure* category focuses on the data structure used to either collect or store information within the Blockchain network, while the *invoke functions* category focuses on the mechanisms used by various organisations (i.e., Local Municipality, District Municipality, Investigator's Firm, and Auditor's Firm) to interact with the ShareTendPro network. Additionally, the *invoke functions* category is classified into three sections namely *InitLedger*, *CreateTender*, and *QueryTender*, as shown in Figure 6.3. The *InitLedger* section contains the detail that seeks to initialise the ledger within the Blockchain network. The *CreateTender* section contains details that seek to add new project information to the

Blockchain network, while the *QueryTender* section focuses on the details that seek to query the data stored within the Blockchain network.

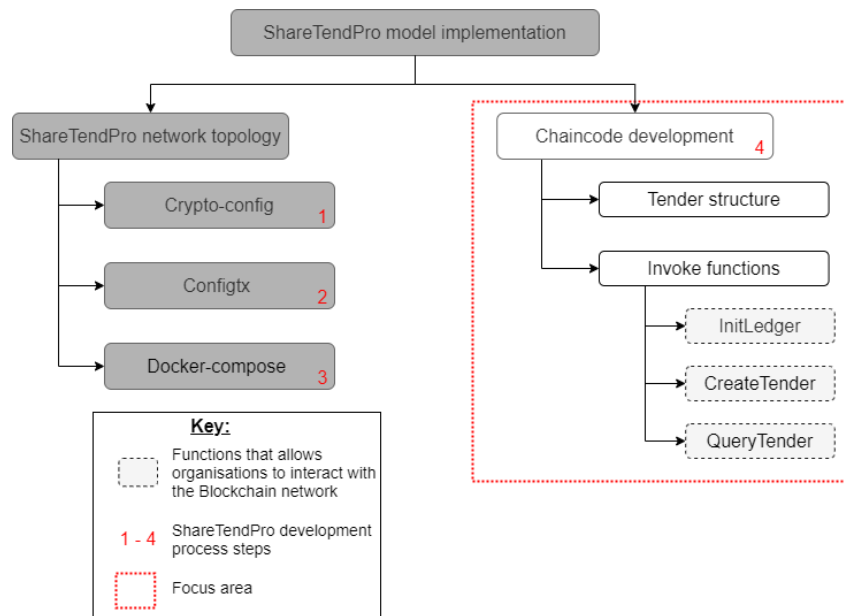


Figure 6.3 Chaincode development as the focus area

6.4 Summary

This chapter explored the implementation of the proposed solution on a high level. The tools used to develop the prototype are divided into two requirements namely hardware and software requirements, while the implementation of the ShareTendPro model is divided into two namely the ShareTendPro network topology and chaincode development. Additionally, the ShareTendPro network topology focused on the implementation of the Blockchain network, while the chaincode development focused on the rules that govern the Blockchain network.

The following chapter delves into the results generated from the proposed solution whilst it presents how the solution works in general.

7. Chapter 7: ShareTendPro prototype results

7.1 Introduction

Chapter 6 of this study discussed the implementation of the ShareTendPro model that might be used to address the identified problem of using paperwork to share tendering project information. Chapter 6 explored the necessary details required to implement the ShareTendPro model which includes the three important files namely: *crypto-config.yaml* file, *configtx.yaml* file, and *docker-compose.yaml* file. Note that Appendix B contains a detailed implementation of Chapter 6, and should Chapter 7 refer to ‘all details in Chapter 6’, such details would rather be contained in Appendix B. All these files explored the configuration details that seek to implement the Blockchain network of the ShareTendPro network. Additionally, Chapter 6 also detailed the implementation of the chaincode used to govern the Blockchain network.

Therefore, this chapter is devoted to the results generated after executing the ShareTendPro model as implemented in the previous chapter. Note that this chapter is bound to the limitations set in the previous chapter, hence, the results contained within this chapter form part of the transactions that take place within the Blockchain network. The remainder of this chapter is structured as follows: a scenario that seeks to visualise how the ShareTendPro network might be used to share project information is detailed, while the following section explores the ShareTendPro results. Thereafter, the last section summarises the entire chapter by providing a conclusion.

7.2 Scenario

The scenario depicted in this section is like scenario 2 that was explored in Chapter 4. However, the only difference is that the scenario contained in this section is more detailed compared to scenario 2 of Chapter 4. Note that this section makes use of this scenario to portray the research problem that would be addressed by the ShareTendPro network later. Furthermore, note that the researcher acknowledges that not all tendering projects would be audited and investigated. However, this scenario depicts a use-case study whereby both Auditors and Investigators were initially involved when it comes to either auditing or investigating the initial phase of the project. Therefore, the process that happens within the scenario is as follows (note that the numbering below corresponds to the numbering in Figure 7.1):

1. The Local Municipality opens tendering project X for bidding.

2. Various suppliers apply for tender project X by submitting tender documents to the Local Municipality.
3. The tendering committee assigned by the Local Municipality assesses all the suppliers who applied for project X and submits the results of the assessment to the Local Municipality. Note that the tender committee are responsible for handling all the issues or fraudulent that emanates from the procurement processes. Some of these issues include related nepotism, falsifying the bid documents in relation to competency area, as well as favouritism by crafting the tender requirements to favour a particular supplier. Hence, these issues falls outside the scope of this study because the proposed solution seek to monitor the execution of the tendering projects, which takes place after the procurement processes.
4. The Local Municipality awards project X to Supplier S based on the outcomes presented by the tendering committee.
5. The Local Municipality assigns Peter to manage project X. Thereafter, Peter uses computer *LM_NO* (which stands for Local Municipality node 0) to issue a progress report for project X as part of his responsibilities which seek to portray the following progress “so far, 20% of project X was completed within four months”.
6. Peter shared this report with John from the Auditor’s Firm who was tasked to audit the financial expenditure of tendering project X. Hence, the project report act as proof of work that was completed by Supplier S during the payments claims that falls under the sample payments reports that were selected by an Auditor.
7. Peter also shared this report with David from the Investigator’s Firm who was tasked to investigate allegations of corruption in tendering project X. The report acts as proof of work completed by Supplier S.
8. Later on, the District Municipality opens tender project Y for bidding. Assume that project Y is similar to project X.
9. Assume that Supplier S decided to collude with Peter when it comes to falsifying the report of project X to portray the following progress “50% of project X was completed within four months”. Note that the researcher acknowledges that they might be a number of players involved to achieve this objective which includes ICT administrators that are responsible for data stored within the repositories and other backup mechanisms. This assumption is drawn from the report presented by the Department of Community Safety during the assessment of the dockets stored in the achieve since they

reported that approximately 63% of the case dockets were lost on the achive system without any disciplinary action [100] [101].

10. Various suppliers apply for project Y, including Supplier S. Assume that Supplier S has included a falsified progress report of project X when applying or bidding for project Y and included Peter as a referee who can provide more clarifications regarding project X.
11. The District Municipality assigns Martha from the tendering committee of project Y a task to request a progress report of project X from Peter as part of trying to confirm whether Supplier S managed to complete 50% of the project within four months or not. Note that Martha used computer *DM_N0* (which stands for District Municipality node 0) to send an electronic email to Peter when requesting the progress report of project X.
12. Peter submitted a falsified progress report of project X to Martha (*DM_N0*) at the District Municipality.
13. The tendering committee of the District Municipality assesses all the suppliers who applied for project Y and submits the results of the assessment to the District Municipality.
14. The District Municipality awards tendering project Y to Supplier S based on the outcome of the assessment which was motivated by the information provided by the supplier and confirmed by Peter who works at the Local Municipality.

Due to the outcome of 14, the District Municipality now trusts the Local Municipality. Assume that the District Municipality finds out later that Supplier S is incompetent when it comes to executing project Y. This happened because the District Municipality was confident enough to trust that Supplier S is competent since it has included someone from the Local Municipality, as their referee.

The main objective of this scenario was to depict a loophole that might be used to tamper with project information in such a way that it can be used to influence the decision of other projects offered by a different municipality. For instance, in the scenario, a falsified report of project X was used to influence the decision when it comes to awarding project Y offered by the District Municipality. Figure 7.1 seeks to visualise this scenario as various people in different organisations interact with either a falsified or a legit report of project X. Assume that the communication mechanism used to share the report of project X was an electronic mail (e-

mail). Hence, Figure 7.1 depicted the computers used by various people in different organisations as they interact with an electronic report of project X.

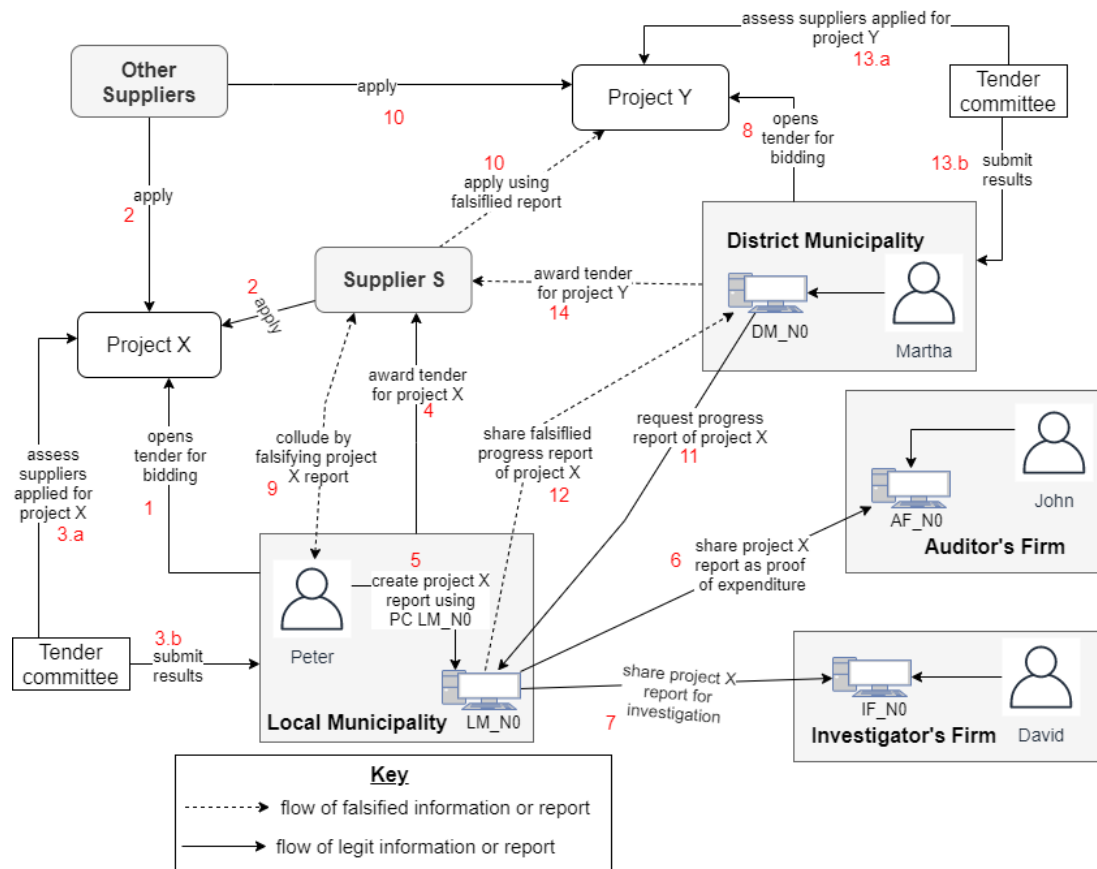


Figure 7.1 Scenario

To support the current tendering system, this study introduced the ShareTendPro network, instead of conventional email, that seeks to connect all the computers of various organisations that have an interest in the tendering project regardless of their geographical location. For instance, the computers that have an interest in project X are *LM_NO* (i.e., Local Municipality node 0), *DM_NO* (i.e., District Municipality node 0), *IF_NO* (i.e., Investigator's Firm node 0), and *AF_NO* (i.e., Auditor's Firm node 0) as shown in Figure 7.1. Therefore, the ShareTendPro network would be used as a tool that replaces e-mail when it comes to sharing project information with all the people that have an interest in the tendering project. Additionally, the establishment of the ShareTendPro network also allows these computers to share project information securely while preserving the integrity of the information. The establishment of the ShareTendPro network as a solution is also aimed at enforcing trust and transparency among various organisations that have an interest in the tendering project.

Figure 7.2 depicts how this study addresses the identified problem within the scenario by introducing the ShareTendPro network as a solution. A more detailed discussion of the ShareTendPro solution as shown in Figure 7.2 follows next in an attempt to solve the problem shown in Figure 7.1.

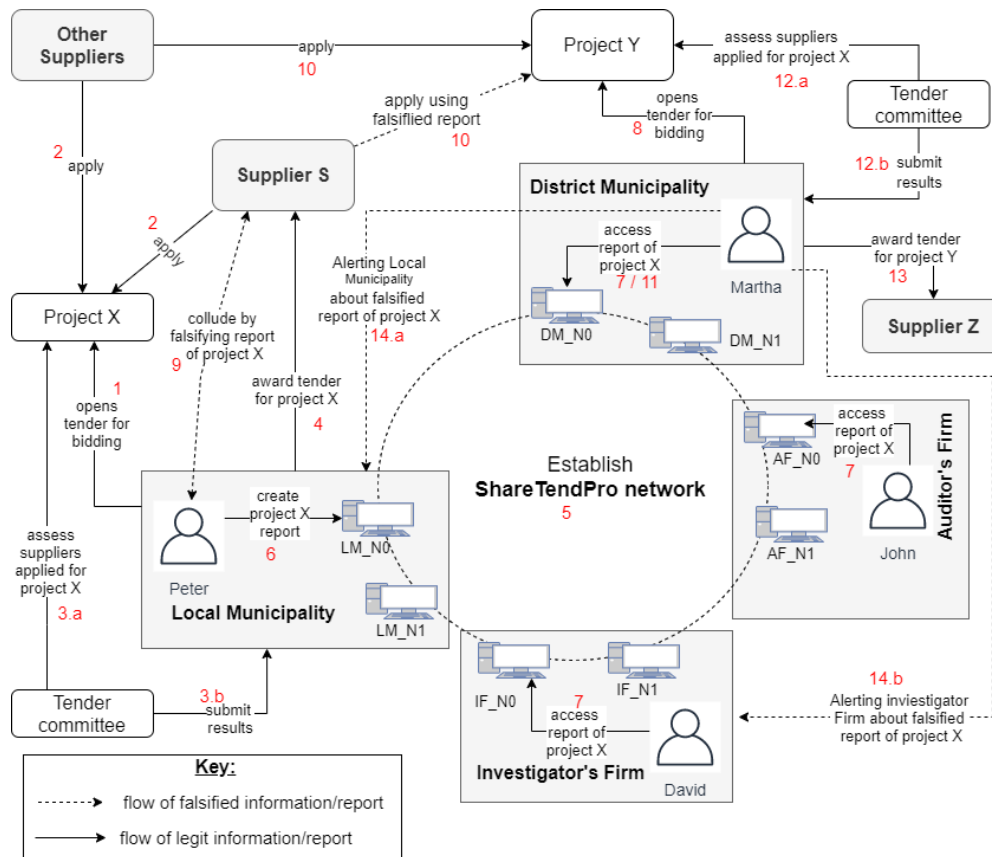


Figure 7.2 ShareTendPro solution

The process that happens within the ShareTendPro solution is as follows:

1. Steps 1-4: these steps are similar to steps 1-4 as discussed in the scenario of Figure 7.1.
5. Step 5: represents the establishment of the ShareTendPro network that would be used to share project information securely while preserving the integrity of the information.
6. Step 6: depicts Peter using computer *LM_NO* to create a progress report of project X. Note that computer *LM_NO* is one of the computers of the Local Municipality that has joined the ShareTendPro network – hence, the report created by Peter would be stored within the blockchain of the ShareTendPro network.
7. Step 7: depicts various computers accessing the report of project X that was created using computer *LM_NO*. Note that this step is automatically activated when computer *LM_NO* submits the report of project X to the Blockchain in the ShareTendPro network,

whereby the ShareTendPro network distributes it to all the computers that have joined the communication channel, due to the inner workings of the blockchain.

8. Step 8: depicts the District Municipality opening project Y for bidding. This step is similar to step 8 of Figure 7.1.
9. Step 9: depicts Supplier S and Peter colluding by falsifying the report of project X. This step is similar to step 9 of Figure 7.1. Later it will become clear how this falsification is detected.
10. Step 10: depicts various suppliers applying for tendering project Y offered by the District Municipality, including Supplier S. This step is similar to step 10 of Figure 7.1. Assume that Supplier S has included the falsified report on the tendering documents when bidding for project Y.
11. Step 11: represents Martha who was tasked by the tendering committee of project Y to confirm the progress report submitted by Supplier S within the ShareTendPro network. Note that Martha at node DM_N0 did not request the report of project X as compared to the scenario depicted in step 11 of Figure 7.1 because the report is now available in the ShareTendPro network (Blockchain) as she can access it directly.
12. Step 12: depicts the tendering committee of the District Municipality assessing all the suppliers that have applied for project Y and submitting the results of the assessment to the District Municipality. However, the tendering committee realised that the report (i.e. document) of project X submitted by Supplier S contradicts the actual details (i.e. the report) stored within the blockchain of the ShareTendPro network. Due to this discrepancy, Supplier S is removed from the bidding process of project Y with consequences, and another supplier will need to be appointed.
13. Step 13: depicts the District Municipality awarding tender project Y to Supplier Z. Note that this was achieved after penalising Supplier S since the information or report provided by the supplier does not correspond with the actual report stored within the ShareTendPro network.
14. Step 14: represents Martha who is part of the tendering committee of project Y alerting the Local Municipality and Investigator's Firm about the falsified report of project X for further investigations. The Local Municipality will conduct an internal investigation to discipline Peter, while the Investigator's Firm will conduct corruption-related activities or investigations between Peter and Supplier S which include acts of bribery. This, however, is out of the scope of this research and will not be shown further.

Note that the implementation of the ShareTendPro network introduced within Figure 7.2 was discussed in the previous chapter and more detail in Appendix B. The following section focuses on the results generated after executing or testing the ShareTendPro network, as the ShareTendPro network results.

7.3 ShareTendPro network results

This section focuses on the results generated during the process of testing the ShareTendPro network. However, the results contained within this section are classified into two categories of processes namely: *establishing Blockchain network* and *interacting with Blockchain network* as shown in Figure 7.3. The first category which is *establishing Blockchain network* focuses on the results generated during the process of establishing a mechanism used by the ShareTendPro network to connect all the virtual nodes (i.e., computers) of various organisations that have an interest in participating in the ShareTendPro network. The second category which is *interacting with Blockchain network* focuses on the results generated by various nodes as they interact with the Blockchain network. Note that number 5 in Figure 7.3 corresponds to number 5 in Figure 7.2, since it seeks to establish the Blockchain network used by the ShareTendPro prototype. Additionally, note that numbers 6, 7, and 11 depicted in Figure 7.3 corresponds to numbers 6, 7, and 11 of Figure 7.2 since they seek to interact with the Blockchain network used by the ShareTendPro prototype. Hence, the *establishing Blockchain network* process contains step 5 of Figure 7.2, while the *interacting with Blockchain network* process contains steps 6, 7, and 11 of Figure 7.2.

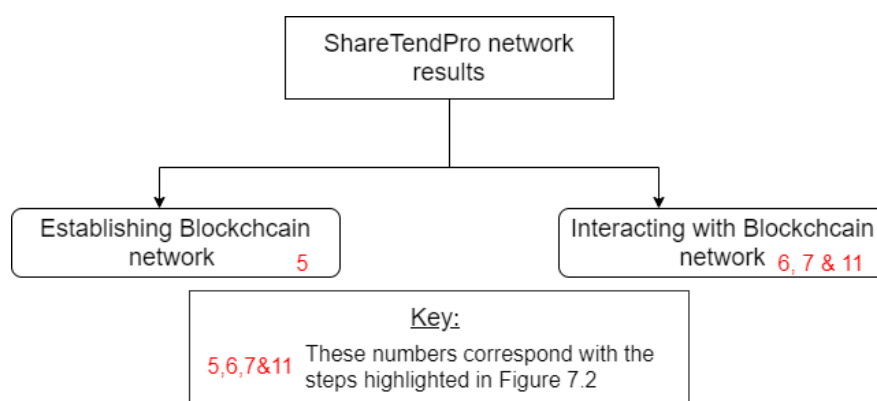


Figure 7.3 High-level of the ShareTendPro network results

The following section explores the results generated during the process of establishing a Blockchain network used by the ShareTendPro prototype.

7.3.1 Establish Blockchain network

This section focuses on the results generated by the process that seeks to establish the Blockchain network used by the ShareTendPro prototype as shown in Figure 7.4. The results contained within this section only focus on getting the Blockchain network up and running, hence, the details of these results are found in Appendix C. The results contained within this section are classified into three subsections namely: *running Blockchain network*, *creating communication channel*, and *deploying smart-contract* (i.e. chaincode) to get the Blockchain network up and running successfully. Again, note that the in-depth details of these results are explored in Appendix C for more details. The first subsection, which is *running Blockchain network* (represented by label 5.a in Figure 7.4) focuses on the results generated during a process of connecting all the virtual nodes of various organisations that has an interest in participating in the ShareTendPro network. The second subsection, which is *creating communication channel* (represented by label 5.b in Figure 7.4) focuses on the results generated during a process of creating a private communication channel used by these virtual nodes to confidentially share project information. The last subsection, which is *deploying smart-contract* (represented by 5.c in Figure 7.4) focuses on the results generated by a process of deploying the rules (also known as smart-contract or chaincode) that governs the Blockchain network used by the ShareTendPro prototype. The deployment process includes *installing*, *approving*, and *committing* the chaincode by various nodes that have joined the Blockchain network.

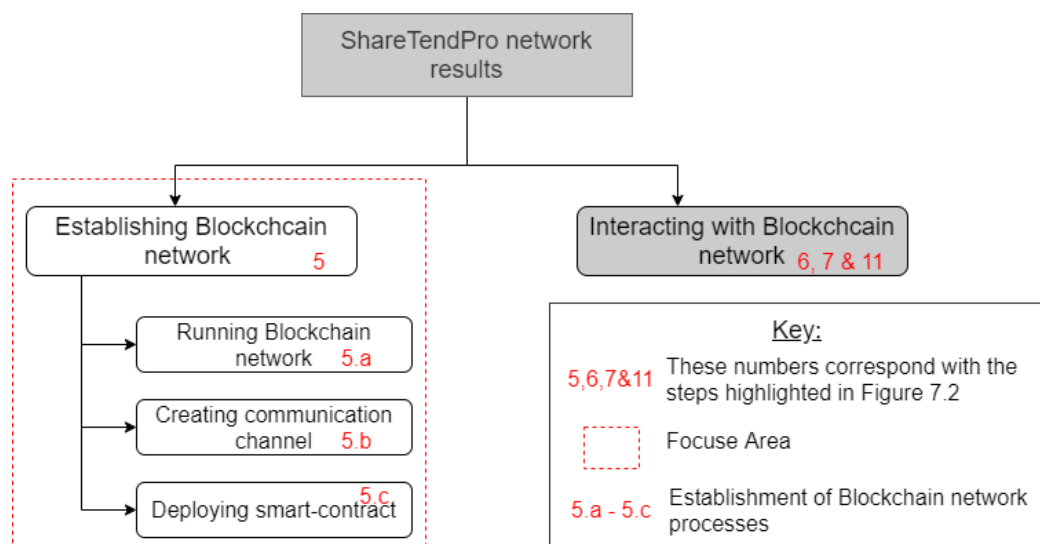


Figure 7.4 Establish Blockchain network as a focus area

After the Blockchain network is up and running, we can proceed to a process that seeks to interact with the network used by the ShareTendPro prototype. Hence, the following section focuses on the results generated by various nodes as they interact with the Blockchain network.

7.3.2 Interacting with Blockchain network

This section focuses on the results generated by various processes that seek to interact with the Blockchain network used by the ShareTendPro prototype, as shown in Figure 7.5. In other words, this section focuses on the results generated by various nodes as they interact with the Blockchain network using various methods or functions contained within the smart-contract (i.e. chaincode). Therefore, the results contained within this section are classified into five subsections namely: *create tender*, *query tender*, *update tender*, *delete tender*, and *query tender history*, as shown in Figure 7.5. The first subsection, which is *create tender* focuses on the results generated by a process that allows a specific node or computer to create a project report of a particular tendering project. Note that the *create tender* subsection represents the results generated by step 6 in Figure 7.2. The second subsection, which is *query tender* focuses on the results generated by a process that allows various nodes to access the project report created using the *create tender* subsection. Note that the *query tender* subsection represents the results generated by step 7 in Figure 7.2. The third subsection, which is *update tender* focuses on the results generated by a process that allows a specific node (i.e., computer) to update the project information of a tender (i.e., project X in Figure 7.2). Updates to the project information may simply include editorial updates or incorporating feedback from stakeholders. The fourth subsection, which is *delete tender* focuses on the results generated by a process that allows a specific node (i.e., computer) to delete the project information of a tender project (i.e., project X in Figure 7.2). Deleting a tender may be used to simply delete the details of a particular tendering project that was awarded to a supplier who might have decided to withdraw or terminate the contract. However, note that even if the details of this tendering project were deleted, the evidence of its previous existence will not be deleted in the Blockchain of the ledger. This evidence is readily available within the ShareTendPro network, and it can be accessed through a process that seeks to query the project history of that tendering project. The last subsection, which is *query tender history* focuses on the results generated by a process that allows a specific node (i.e. computer) to access the project history of a tendering project. Note that the *query tender history* can be accessed by any node that has joined the channel, however, note that this process is represented by step 11 in Figure 7.2.

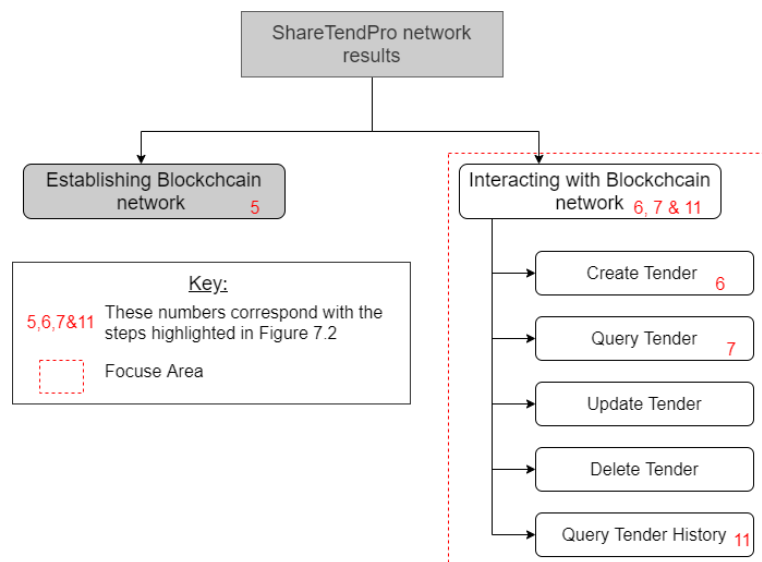


Figure 7.5 Interacting with Blockchain network as a focus area

7.3.2.1 Create tender

This section focuses on the results generated by step 6 of Figure 7.2 and as shown in Figure 7.5, which is the creation of a project report by computer LM_N0. Therefore, line 202-212 of Figure 7.6 depicts a function called *chaincodeInvokeByAddingProject()* that contained a command line used by peer0 (i.e. corresponding to LM_N0) of the Local Municipality to create the tendering project information of project X. Line 203 seeks to call a function called *setGlobalsForPeer0LocalMunicipality()* that allows the Blockchain network to initialise peer0 (LM_N0) in the Local Municipality. Lines 204-211 represent the command line that seeks to create tendering project information to the Blockchain network. Line 210 represents a flag that seeks to execute a function called *createTender()* contained within the smart-contract with the following five arguments namely:

- $args[0] = \text{"Tender1000"}$
- $args[1] = \text{"Local Municipality"}$
- $args[2] = \text{"Project X"}$
- $args[3] = \text{"20% of the project has been completed within four months"}$
- $args[4] = \text{"Supplier S"}$


Note that this project report is identified using the key "Tender1000". Label R of Figure 7.6 represents the results displayed within the command prompt after executing the *chaincodeInvoke()* function, as presented by line 213. Additionally, the command line returns

a successful status of 200 which implies that the project report of the tendering project X was created or added to the Blockchain network successfully, as shown in label R.

```

202 chaincodeInvokeByAddingProject() {
203   setGlobalsForPeer0LocalMunicipality
204   peer chaincode invoke -o localhost:7050 --ordererTLShostnameOverride orderer1.ordererOrg.workplace \
205     --tls $SCORE_PEER_TLS_ENABLED --cafile $ORDERER_CA -C $CHANNEL_NAME -n ${CC_NAME} \
206     --peerAddresses localhost:7051 --tlsRootCertFiles $PEER0_LocalMun_CA \
207     --peerAddresses localhost:9051 --tlsRootCertFiles $PEER0_DistrictMun_CA \
208     --peerAddresses localhost:11051 --tlsRootCertFiles $PEER0_InvestigatorFir_CA \
209     --peerAddresses localhost:13051 --tlsRootCertFiles $PEER0_AuditorFir_CA \
210     -c '{"function": "createTender","Args":["Tender1000", "Local Municipality", "Project X",
211         "20% of the project has been completed within four months", "Supplier S"]}'
212 }
213 chaincodeInvokeByAddingProject

```



```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
2021-09-14 20:02:53.120 SAST [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200 payload:"
{"tenderOwner\":\"Local Municipality\",\"projectName\":\"Project X\",\"reports\":\"20% of the project has been completed within four mo
nths\",\"supplier\":\"Supplier S\"}"
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$

```

Figure 7.6 Create Tender

After the project information has been added to the Blockchain network, that information would automatically be available for other nodes to access it, as long as they are on the same network. The following section focuses on the results generated by the *query tender* process. Note that other nodes use the key “Tender1000” to access the project report created in Figure 7.6.

7.3.2.2 Query tender

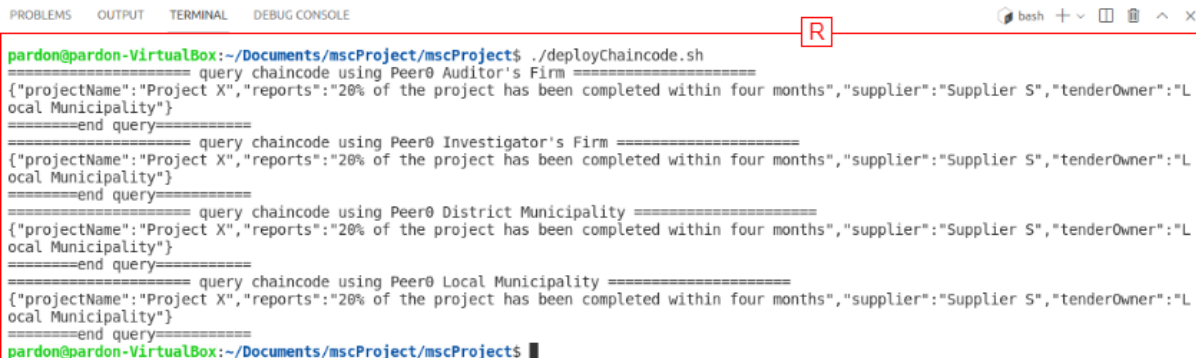
This section focuses on the results generated by step 7 of Figure 7.2 and as shown in Figure 7.5, which is a process used by other nodes to access project report (using “Tender1000”) of tendering project X that was created in Figure 7.6. Note that the results contained within this section depict the process represented by step 7 in Figure 7.2. Therefore, lines 215-232 of Figure 7.7 depict the details of a function called *chaincodeQuery()*, which contains the command lines used by various nodes to access the project report of tendering project X. For instance, lines 217 and 218 represent the command that allows peer0 of the Auditor’s Firm to access the project report of tendering project X (using “Tender1000”). Additionally, line 2017 seeks to call a function called *setGlobalsForPeer0AuditorFirm()* that allows the Blockchain network to initialise peer0 (i.e. AF_N0) in the Auditor’s Firm, while line 218 depicts the command line used by peer0 of the Auditor’s Firm to access the report of tendering project X using the key “Tender1000”. Note that the command line makes use of a function called *queryTender()* contained within the chaincode to interact with the ShareTendPro network, as shown in Figure 7.7. The results of this command line are displayed within the command prompt represented by label R. However, the same notion applied to the command line represented in line 218 can also be used to other command lines used by other nodes. Label R

of Figure 7.7 represents the results generated after executing the `chaincodeQuery()` function using line 233. Hence, the results contained within this section also portray the distributed nature of the ShareTendPro network because other nodes automatically have access to the project report of tendering project X compared to the scenario depicted in Figure 7.7.

```

215 chaincodeQueryByProject() {
216     echo "===== query chaincode using Peer0 Auditor's Firm ===== "
217     setGlobalsForPeer0AuditorFirm
218     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} -c '{"function": "queryTender","Args":["Tender1000"]}'
219     echo "=====end query===== "
220     echo "===== query chaincode using Peer0 Investigator's Firm ===== "
221     setGlobalsForPeer0InvestigatorFirm
222     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} -c '{"function": "queryTender","Args":["Tender1000"]}'
223     echo "=====end query===== "
224     echo "===== query chaincode using Peer0 District Municipality ===== "
225     setGlobalsForPeer0DistrictMunicipality
226     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} -c '{"function": "queryTender","Args":["Tender1000"]}'
227     echo "=====end query===== "
228     echo "===== query chaincode using Peer0 Local Municipality ===== "
229     setGlobalsForPeer0LocalMunicipality
230     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} -c '{"function": "queryTender","Args":["Tender1000"]}'
231     echo "=====end query===== "
232 }
233 chaincodeQueryByProject

```



```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
===== query chaincode using Peer0 Auditor's Firm =====
{"projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier S", "tenderOwner": "Local Municipality"}
=====end query=====
===== query chaincode using Peer0 Investigator's Firm =====
{"projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier S", "tenderOwner": "Local Municipality"}
=====end query=====
===== query chaincode using Peer0 District Municipality =====
{"projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier S", "tenderOwner": "Local Municipality"}
=====end query=====
===== query chaincode using Peer0 Local Municipality =====
{"projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier S", "tenderOwner": "Local Municipality"}
=====end query=====
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ █

```

Figure 7.7 Query Tender

After the project information of tendering project X is distributed to other nodes, we can then proceed with a process that seeks to update some of the details contained within the project report. The following section focuses on the results generated by a process that seeks to update the details of the supplier and project report. Note that this process can only be used by nodes that joined the communication channel. Hence, the update process might still be used for illegal altering of project information. However, the project history that would be discussed later can be used as a mechanism that seeks to explore what has transpired within that project since the information stored within the ShareTendPro network is immutable by default. This implies that the information stored within the transaction logs or Blockchain of the ledger cannot be deleted, however, the information stored within the world-state of the ledger can be changed or updated as the network transition from one state to the other.

7.3.2.3 Update tender

This section focuses on the results generated by a process that can be used to update the project information stored within the ShareTendPro network. However, this section explores the two possible updating processes implemented within the ShareTendPro network which are “*update tender supplier*” and “*update tender report*”. The first process, which is to *update the tender supplier* focuses on the results generated during a process that seeks to update the details of the supplier (e.g., contact details or to assign a new supplier altogether). The second process, which is the *update tender report* focuses on the results generated by a process that seeks to update the report of the tendering project (e.g., doing editorial updates).

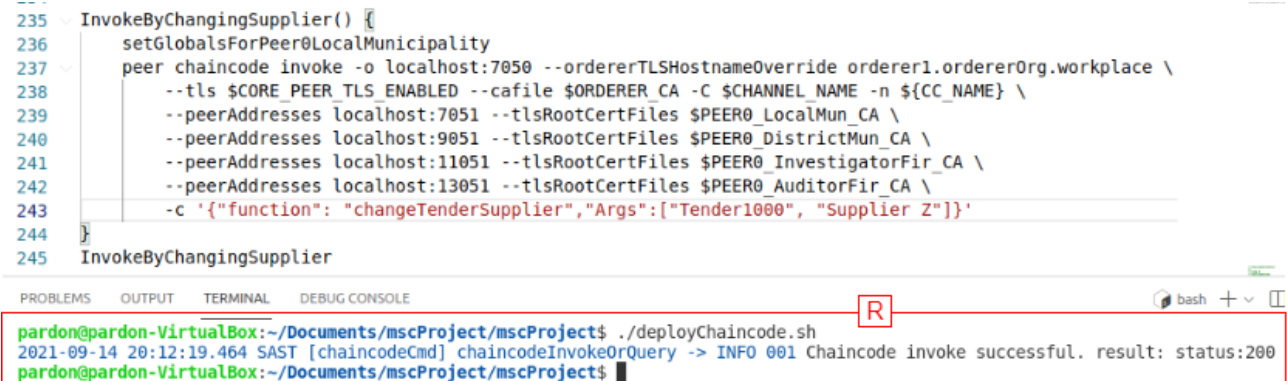
a) Update Tender Supplier

This section focuses on the results generated by a process that seeks to update supplier information on tendering project X. For instance, lines 235-244 of Figure 7.8 depict a function called *InvokeByChangingSupplier()*, which is a function used by peer0 (i.e., computer LM_N0) of the Local Municipality to update the details of a supplier from “Supplier S” to “Supplier Z”. As indicated in lines 237-243, the command makes use of a function called *changeTenderSupplier()*, as seen in line 243, contained within the chaincode to interact with the ShareTendPro network. The *changeTenderSupplier()* function requires two arguments. The first argument represents a key “Tender1000” is used to identify a tendering project, while the second argument is used to assign the updated details of the supplier which is “Supplier Z” in this case. Therefore, the results displayed (as presented by label R) are generated after executing the *InvokeByChangingSupplier()* function using line 245. The status of the results reflected within label R is 200, which implies that the supplier details were updated successfully.

```

235 InvokeByChangingSupplier() {
236     setGlobalsForPeer0LocalMunicipality
237     peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
238         --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA -C $CHANNEL_NAME -n ${CC_NAME} \
239         --peerAddresses localhost:7051 --tlsRootCertFiles $PEER0_LocalMun_CA \
240         --peerAddresses localhost:9051 --tlsRootCertFiles $PEER0_DistrictMun_CA \
241         --peerAddresses localhost:11051 --tlsRootCertFiles $PEER0_InvestigatorFir_CA \
242         --peerAddresses localhost:13051 --tlsRootCertFiles $PEER0_AuditorFir_CA \
243         -c '{"function": "changeTenderSupplier", "Args":["Tender1000", "Supplier Z"]}'
244 }
245 InvokeByChangingSupplier

```



```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
2021-09-14 20:12:19.464 SAST [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$

```

Figure 7.8 Change Tender Supplier

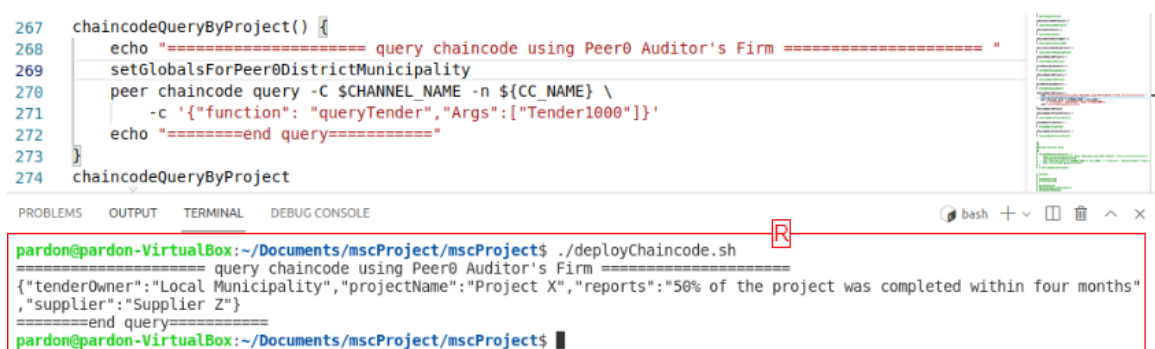
After updating the details of the supplier, the second process that seeks to update the details of a project report is considered. Hence, the following section explores the results generated by a process that seeks to update the project report of a tendering project.

b) Update Tender Report

This section explores the details of the results generated by a process that seeks to update the details of the project report of tendering project X. However, note that the results generated using the update tender report process are similar to the results generated by the update tender supplier process (as depicted in Figure 7.8). Hence, the in-depth details of the results generated by the update tender report process are explored in Appendix C for more details. Furthermore, note that the update tender report process seeks to change the project report to reflect the new report that portrays the following progress “50% of the project was completed within four months”.

To verify whether indeed the project information (which is both supplier and report details) was updated successfully within the ShareTendPro network, then we can use a *query tender* process to access the latest updates of the tendering project X using the key “Tender1000”. Therefore, Figure 7.9 depicts the results generated by the *query tender* process. Note that the project information was successfully updated since it has changed:

- **From:** “20% of the project was completed within four months”, as shown in Figure 7.7.
To: “50% of the project was completed within four months”, as depicted in Figure 7.9.
- **From:** “Supplier S”, as presented in Figure 7.7.
To: “Supplier Z”, as shown in Figure 7.9.



```

267 chaincodeQueryByProject() {
268   echo "===== query chaincode using Peer0 Auditor's Firm ===== "
269   setGlobalsForPeer0DistrictMunicipality
270   peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} \
271     -c '{"function": "queryTender","Args":["Tender1000"]}'
272   echo "=====end query===== "
273 }
274 chaincodeQueryByProject

```

```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
===== query chaincode using Peer0 Auditor's Firm =====
{"tenderOwner":"Local Municipality","projectName":"Project X","reports":"50% of the project was completed within four months",
,"supplier":"Supplier Z"}
=====end query=====
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$

```

Figure 7.9 Query Changed Tender Report

After updating the details of a tendering project, one might decide to delete the project information stored within the ShareTendPro network. Assume that this process was activated by a withdrawal of the tendering project by a supplier that results in the termination of a contract between “Supplier Z” and the Local Municipality. Therefore, the following section

explores the results generated by a process that seeks to delete project information stored within the ShareTendPro network. As highlighted in the previous section, the evidence of the existence of tendering project X will not be completely lost because this information is still stored in the transaction logs or Blockchain of the ledger and it is immutable by default (which implies that it cannot be changed or permanently deleted for historic investigation purposes).

7.3.2.4 Delete tender

This section focuses on the results generated by a process that seeks to delete project information from the ShareTendPro network. However, note that the results generated using the delete tender process are similar to the results generated by the update tender supplier process (as depicted in Figure 7.8). Hence, the in-depth details of the results generated by the delete tender process are explored in Appendix C for more details. Furthermore, note that the deleted information can only be accessible using a process that seeks to *query the project history* of that tendering project.

After performing all the processes that seek to either create, update or delete the project information of a particular tendering project, then the evidence related to such processes is stored within the transaction logs. The information stored within the transaction logs is immutable since it seeks to preserve the evidence of what transpired within that specific tendering project. Additionally, the evidence contained within the transactional logs can also be used as forensic data since it portrays the entire history of a particular tendering project. Therefore, the following section focuses on the results generated by a process that seeks to access a project history of tendering project X.

7.3.2.5 Query tender history

This section explores the results generated by a process that seeks to access the project history of a particular tendering project, which is project X. Therefore, lines 296-302 of Figure 7.10 depicts a function called *chaincodeQueryProjectHistory()*, which contains the details used by a command line that seeks to access the project history of a tendering project associated with the key “Tender1000”. As indicated in lines 299 and 300, the command makes use of a function called *getHistoryForTender()*, as seen in line 300, contained within the chaincode to interact with the ShareTendPro network. The results displayed (as presented by label R) are generated after executing the *chaincodeQueryProjectHistory()* function using line 303. Note that the project information displayed within label R contains various transaction IDs and each of these

transaction IDs represents a specific process that was executed using a key “Tender1000”. Note that a number of these processes have been executed already, as follows. The processes that were executed using a key “Tender1000” are *creating a Tender project* (as shown in Figure 7.6), *updating the tender supplier* (as shown in Figure 7.8), *updating the tender report* (as shown in Figure C.12), and *deleting a tender project* (as shown in Figure C.13). All these processes are identified using a unique transaction ID (represented by *TxID* in label R). Again, note that the status of a tag “IsDelete” is only true when it comes to the Transaction ID that represents a process that seeks to delete the information of a tendering project, else it is false.

```

296 chaincodeQueryProjectHistory() {
297     echo "===== query chaincode using Peer0 Auditor's Firm ====="
298     setGlobalsForPeer0DistrictMunicipality
299     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} \
300     -c '{"function": "getHistoryForTender", "Args":["Tender1000"]}'
301     echo "=====end query===== "
302 }
303 chaincodeQueryProjectHistory

```

```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProjects$ ./deployChaincode.sh
===== query chaincode using Peer0 Auditor's Firm =====
[{"TxId": "a35fa2ec5f2225a5befc52119e1e81d7fdbdc2b8e369281e15f227748c6781681e9e9c", "Value": null, "Timestamp": "2021-09-15 12:28:09.409964752 +0000 UTC", "IsDelete": "true"}, {"TxId": "919896445f5350fbeb11c3011140090b5219d9593b9fd7643dac91ed6d31030f", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "50% of the project was completed within four months", "supplier": "Supplier Z"}, "Timestamp": "2021-09-15 12:24:46.098319371 +0000 UTC", "IsDelete": "false"}, {"TxId": "b31185f27ae9cd52119e1e81d7fdbdc2b8e369281e15f227748c6781681e9e9c", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier Z"}, "Timestamp": "2021-09-15 12:23:23.951643768 +0000 UTC", "IsDelete": "false"}, {"TxId": "94a208cc045ee21011580be40f2241e528381eeb59f78b505a1a75517da258c", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier S"}, "Timestamp": "2021-09-15 12:21:58.628973647 +0000 UTC", "IsDelete": "false"}]
=====end query=====
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProjects$

```

Figure 7.10 Get Tender History

7.4 Conclusion

This chapter explored the results generated by various processes that seek to interact with the Blockchain network used by the ShareTendPro prototype. These processes include *creating a tender project*, *updating the project report* of a particular tendering project, *deleting a particular tendering project*, and *querying the project history* of a specific tendering project. Additionally, this chapter demonstrated the distributed nature of the proposed solution because the project information was shared across different organisations or domains. For instance, a tendering project was created using a computer that belongs to *localMun.workplace* domain and this information were accessed by other computers that belong to other domains of different organisations (i.e., *districtMun.workplace*, *investigatorFir.workplace*, and *auditorFir.workplace*).

This chapter further explored how the ShareTendPro network preserves the evidence of the tendering project by having immutable records of all the transactions that seek to either create, update, or delete project information within the ShareTendPro network.

The following chapter seeks to evaluate the research study. The evaluation criteria used by this study are based on the benefits that were identified during the research including some of the shortcomings that might be encountered when using the ShareTendPro network.

8. Chapter 8: Critical Evaluation

8.1 Introduction

The previous chapter demonstrated how the ShareTendPro network works by exploring the results of various processes associated with it. These processes include *creating tender*, *querying tender*, *updating tender*, *deleting tender*, and *querying tender history*. All these processes allow various computers to interact with the Blockchain network used by the ShareTendPro network.

Therefore, this chapter seeks to evaluate various aspects of this research by analysing whether the research conducted addresses the objectives outlined in Chapter 1. As indicated in Chapter 1, one of the objectives of this research is to implement a Blockchain prototype that might be used to securely and efficiently sharing of project information with the various organisations that have an interest in the tendering project. This research was motivated by using paperwork to share project information because it contributes towards illicit altering of project information during the process. This research was also motivated by several news articles and court cases that are related to the corruptions relegations taking place within the tendering systems or projects. All these factors affect the fairness, transparency, data integrity, and competitiveness of the tendering system. Hence, this study implemented a ShareTendPro network prototype as proof of concept to share project information among various computers that form part of the network regardless of their geographical locations.

The remainder of this chapter is structured as follows: the benefits of this research are discussed in the following section. The shortcomings of this research, which were realised while conducting the research, are detailed in the section to follow, while the last section seeks to summarise the chapter by providing a conclusion.

8.2 Benefits of the research

This section focuses on several benefits associated with this research. However, this section classified these benefits into two categories namely primary benefits and secondary benefits. The first subsection focuses on the primary benefits, which are the benefits that were identified as the results of this research, i.e., specifically on the benefits of the ShareTendPro prototype. The second subsection focuses on the secondary benefits, which are the benefits that are

experienced by users or participants as they interact with project information using the ShareTendPro prototype.

8.2.1 Primary benefits

The primary benefits that were identified by this research study are:

- Distributed nature of the ShareTendPro prototype
- Enhanced security within the ShareTendPro prototype
- Greater transparency over project information
- Automated transactions

Therefore, the following sections explore the details of each of these primary benefits.

8.2.1.1 Distributed nature of the ShareTendPro network

The distributed nature used by the ShareTendPro network refers to the use of a distributed ledger component to share (distribute) project information to all the participants (organisations) that have an interest in the tendering project. The use of a distributed ledger by the ShareTendPro network eliminates issues that emanate from having an organisation that has central powers over project information when it comes to storing or maintaining project reports of a specific tendering project. This implies that they would not be able to collude among participants over project information because all the participants of the proposed solution would have access to the same data. As indicated in Chapter 6, all the computers (virtual nodes) that form part of the ShareTendPro network ensure that it maintains data integrity across the network. The previous chapter also demonstrated the distributed nature of the ShareTendPro network by sharing project information across different domains (*localMun.workplace*, *districtMun.workplace*, *auditorFir.workplace*, and *investigatorFir.workplace*). For instance, peer0 (computer LM_N0) of the Local Municipality created a project report that was automatically distributed or accessed by other nodes that belong to different organisations (domains) within the ShareTendPro network.

After distributing project information among various computers within the network, one might have concerns regarding information security during the process of either sharing or storing project information within the ShareTendPro network. Hence, the following section focuses on the enhanced information security that seeks to secure the project information within the ShareTendPro network.

8.2.1.2 Enhanced information security within the ShareTendPro prototype

Enhanced information security refers to the components used to secure project information within the ShareTendPro network. In other words, the enhanced security focuses on the mechanisms used to increase the difficulty for unauthorised parties (i.e., hackers) to either access or tamper with project information within the ShareTendPro network. The mechanisms used by the ShareTendPro network to secure project information are as follows:

- Project information is distributed to multiple locations, which makes it difficult for hackers to compromise it because the Blockchain network used by the ShareTendPro solution can function with minimum fault tolerance, as indicated in Chapter 3. Another aspect where fault tolerance is achieved, as indicated in Chapter 6 with the Raft ordering service. These mechanisms are derived from the previous benefits due to the distributed nature of the ShareTendPro solution. These mechanisms, therefore, improve the availability of the ShareTendPro network.
- The ShareTendPro network consists of two mechanisms that can be used to improve its confidentiality. The first mechanism focuses on the use of a secure communication channel to restrict access to the project information, while the second mechanism focuses on the use of cryptography to encrypt transactions of the project information within the ShareTendPro network. (As indicated in Figure 7.10, all the transaction IDs are encrypted.)
- Transactions of the project information are timestamped, as indicated in Figure 7.10. All the transactions are assigned a date and time on which it was created. This mechanism improves the integrity of the project information within the ShareTendPro network.
- Transactions stored within the transaction log or Blockchain of the ledger are immutable, which implies that they cannot be altered or deleted, as shown in the previous chapter. This mechanism improves the integrity of the ShareTendPro network.

Hence, enhanced information security is one of the benefits of this study since it addresses the confidentiality, integrity, and availability of the ShareTendPro network. However, having enhanced information security in place does not particularly address the issues of transparency within the ShareTendPro network. Therefore, the following section focuses on the details related to a benefit of having greater transparency over project information within the ShareTendPro network.

8.2.1.3 Greater transparency over project information

Greater transparency over project information refers to a component that seeks to ensure that the business process used to store project information within the ShareTendPro network is transparent. The transparency over the project information within the ShareTendPro network is derived by using a distributed ledger to maintain the data integrity of the ledger that records immutable data within the Blockchain. The distributed ledger ensures that every participant is responsible for storing and maintaining the data stored within the ShareTendPro network, while the Blockchain of the ledger ensures that it records the project history of a tendering project within the ShareTendPro network. Having access to the entire project history of a tendering project enables the participants to have full transparency of what transpired within that tendering project. Hence, having greater transparency over the processes that seek to store project information is one of the benefits identified by this study, since it disables rogue parties' ability to collude.

After discussing the distributed nature of the ShareTendPro network, enhanced security, and having greater transparency over tendering projects, one might be concerned about the performance of the ShareTendPro network when it comes to processing projects information. Therefore, the following section explores a benefit that details the automation of some of the processes or transactions within the proposed solution to increase the efficiency of the Blockchain network used by the ShareTendPro solution.

8.2.1.4 Automated transactions

Automated transactions refer to the use of smart-contract (chaincode) components to reduce human interactions within the ShareTendPro network while increasing the efficiency and speed in which the ShareTendPro network processes its transactions. As indicated in the previous chapter, all the transactions submitted by various computers are sent to a specific method contained within the smart-contract. Thereafter, the processes that follow next are triggered automatically since the smart-contract uses the submitted transactions to perform certain tasks within the ShareTendPro network. For instance, computer LM_N0 in the previous chapter submitted a transaction to create a tendering project to a method called *createTender()* contained within the smart-contract. Thereafter, the smart-contract automatically executed a process that seeks to create a tendering project within the Blockchain network and return the results to the computer that submitted the transaction. When the process of creating a tendering

project is completed, another process is initiated automatically, which is to distribute the project information that was created to other computers that form part of the ShareTendPro network. Hence, the transaction that seeks to create tendering project information within the ShareTendPro network consists of two automated processes, the first process is automatically executed by the smart-contract component and the second process is automatically executed by a distributed ledger component. Therefore, automated transactions form part of the benefits identified by this research study since it seeks to minimise human interactions within the ShareTendPro network.

This section has explored the primary benefits of the ShareTendPro network. Therefore, the following section focuses on the secondary benefits that were identified by this research study.

8.2.2 Secondary benefits

This section focuses on the secondary benefits experienced mostly by users or participants of the ShareTendPro network as they used the Blockchain network of the ShareTendPro network to share project information. Therefore, the six secondary benefits that were identified by this study are:

- Time efficiency
- Reduction of cost
- Credible evidence
- Promotes real-time auditing and investigations
- Enforces trust
- Improved collaboration

The following sections explore each of these secondary benefits in more detail.

8.2.2.1 Time-efficient

Time efficiency can be viewed as the time it takes for a participant to have full access to the entire project history of a specific tendering project of their interest using the ShareTendPro network. Note that the authorised participants within the ShareTendPro network have full access to the data collected by the Blockchain of the ledger since it contains records of the transactions that seek to create, update, or delete project information of a specific tendering project. Therefore, this data is readily available to all the authorised participants who have an interest in knowing what happens within that tendering project. For instance, having access to

such data by the Auditor's Firm enables them to audit the tendering project in real-time, which reduces the time taken to look at the financial documents that portray the expenditure incurred during the execution of the tendering project at the end of a financial year or closing period of the desired project. Furthermore, having access to such data by the Investigator's Firm enables them to conduct a real-time investigation without going to the municipality to collect such information in person. Additionally, having access to such data by both the Auditor's Firm and Investigator's Firm enables them to quickly resolve issues related to irregular expenditure or corruption allegations while promoting accountability on other hand.

The time-efficiency benefits can lead to the reduction of the cost that might be experienced by other participants within the ShareTendPro network who has an interest in the tendering project. Hence, the following section focuses on the reduction of cost benefits.

8.2.2.2 Reduction of cost

Note that the previous section also highlighted the cost, however, the cost in the previous section refers to money the municipality loses due to fraud and corruption, or irregular expenditure, while the cost discussed within this section refers to money saved by other organisations (such as Auditor's Firm and Investigator's Firm) due to time efficiency. Hence, the reduction cost explored within this section is a direct result of time efficiency, and it refers to the cost incurred during the process of collecting project history or evidence of a tendering project. This benefit applies to a participant who has an interest in establishing what transpired within a tendering project. As indicated in the previous chapters, the Blockchain component of the distributed ledger is responsible for storing the project history and this information is readily available on the ShareTendPro network for all authorised parties to access it. For instance, the use of the ShareTendPro network by the investigators enables them to reduce the transportation costs that might be incurred during the process of going to the municipality to collect project information as evidence that can be presented to the court of law. Furthermore, the use of ShareTendPro by the auditors also enables them to reduce the cost incurred during the process of collecting data from the municipality that can be used as proof of work completed by a specific supplier as supporting information for their expenditure.

The reduction of cost leads to not having physical data or evidence that would be collected directly from the municipality or site where the tendering project is located. Therefore, the ShareTendPro network presents the project information as digital evidence, hence, the

following section focuses on the benefit that seeks to explore credible evidence within the ShareTendPro network.

8.2.2.3 Credible evidence

Credible evidence refers to information that can be presented to the court of law as evidence of what has transpired within a specific tendering project. This evidence is collected by an investigator from the ShareTendPro network by accessing the project history of a particular tendering project because the data stored within the Blockchain component of the ledger is immutable, timestamped, and it is cryptographically encrypted (as presented by transaction IDs in Figure 7.10). All the mechanisms of having data that is immutable, time-stamped, and cryptographically encrypted seek to preserve the integrity of the evidence that portrays what transpired within that project. The credible evidence can also benefit other participants that have an interest in tendering projects by using the data stored within Blockchain of the ledger as proof of evidence when presenting their reports (i.e., auditor's report).

Having access to real-time data that is reliable and considered as credible evidence, we can use this data or project information to promote real-time auditing or investigation as part of preserving greater accountability within the tendering system. Therefore, the following section focuses on the benefit of exploring the promotion of real-time auditing and investigation.

8.2.2.4 Promotes real-time auditing and investigations

The promotion of real-time auditing and investigation refers to information that can be accessed in real-time that allows both auditors or investigators to either audit or investigate a tendering project in real-time. The benefit of having real-time auditing or investigation is derived from the distributed nature of the ShareTendPro network because it enables both auditors and investigators to access project information in real-time (as soon as it is stored within the Blockchain network). Having access to the real-time data allows the auditors or investigators to complete their auditing or investigation process quickly, which also contributes towards having greater accountability over the tendering project while aiming at saving the municipality's funds that might be spent through irregular expenditure or corruption activities.

Having access to real-time data promotes accountability by conducting real-time auditing and investigation and trust among the participants within the ShareTendPro network would be strengthened automatically. Therefore, the following section explores the details of a benefit that seeks to enforce trust within the ShareTendPro solution.

8.2.2.5 Enforces trust

The enforcement of trust refers to having more trust in the legitimacy and integrity of the data or project information stored within the ShareTendPro network by all parties involved. This does not imply that the participants of the ShareTendPro network do not trust each other, however, they do not have to because the ShareTendPro network inherently has trust built into the network. This implies that it consists of mechanisms that seek to promote transparency, accountability, data integrity, and securing project information by having immutable data that reflect the entire history of the tendering project. Hence, the participants of the ShareTendPro solution will have more trust in the project information stored within the Blockchain network without having to worry about the illegal altering of project information that might be processed unnoticed. After enforcing trust among various participants within the ShareTendPro network then we can look at how the ShareTendPro network can be used to improve collaboration among them.

8.2.2.6 Improved collaboration

Improved collaboration refers to the use of the ShareTendPro network to share project information among various participants working on the same tendering project. For instance, the improved collaboration can benefit both the Local Municipality and District Municipality when it comes to executing a joint project because the use of the ShareTendPro network will eliminate issues related to collusion over project information. The responsibility of the Local Municipality within the joint project might be executing the tendering project, while the responsibility of the District Municipality might be to oversee the tendering project executed by the Local Municipality. Hence, the use of a ShareTendPro network by these municipalities will ensure that accountability is achieved because the process of reporting that tendering project would be more transparent compared to the conventional method of sharing project information in a joint project.

This section has explored all the benefits that were identified by this research study. Therefore, the following section focuses on the shortcoming of the research study that might be anticipated.

8.3 Shortcomings of the research

Sharing project information using the ShareTendPro network can be a challenging task due to the complexity of the underlying technology (which is Blockchain technology) used by the Blockchain network of the ShareTendPro network. Therefore, the following items explore some of the shortcomings that were identified by this research study:

- Storage constraints
- Off-chain data storage
- Scalability
- Lack of political will

The following sections explore the details of each of these shortcomings.

8.3.1 Storage constraints

One of the foreseeable shortfalls is the storage constraints because the project information stored within the Blockchain network is distributed to all the computers that form part of the ShareTendPro network. As indicated in Chapter 6, all the computers contain the same database (CouchDB), hence, the deployment of the ShareTendPro network might require more storage to cater to a large volume of data that would be shared among various participants. Currently, the storage requirements are not that large as, typically, the size of the database is in the order of megabytes. However, as such a network might grow in the future, storage requirements might increase drastically. Therefore, having a computer with limited storage might have a tremendous impact on the effectiveness of that computer to participate in the ShareTendPro network, because its storage might be used up quickly, which ends up rendering that computer inactive. Again, at the moment the storage requirements are low, but in future, the situation might require more storage. Therefore, this shortfall on how to address such issues could warrant further research.

The storage constraints might raise issues related to storing project information whenever a particular computer becomes inactive due to either a power outage or internet connection issues. Hence, the following section explores the off-chain data storage that might be considered as a shortfall that needs to be addressed.

8.3.2 Off-chain data storage

Another shortfall encountered during this research is the need for a temporary storage mechanism that can be used by an offline or inactive computer to store project information and upload it to the ShareTendPro network once that computer becomes active. Some of the issues that might contribute towards the need of having such a component on the ShareTendPro network might be issues that emanate from having an interrupted internet connection or unforeseeable power outage issues. Therefore, this shortfall warrants further research in processing offline data or project information.

Storage constraints and off-chain data storage might raise issues related to the scalability of the ShareTendPro network when it comes to analysing data that was collected by the Blockchain network of the ShareTendPro network. Hence, the following section focuses on the scalability that might need to be considered when it comes to analysing project information stored within the ShareTendPro network.

8.3.3 Scalability

The ShareTendPro network is scalable since it uses a distributed ledger and a secure communication channel. However, this study does not place attention on the factors that are related to analysing the data collected by the ShareTendPro network since it could result in having a large volume of data that might need to be processed. This shortfall could have an impact on the Blockchain network because it might affect the efficiency of the ShareTendPro network in general. Therefore, this shortfall could warrant future research since it will fall under the direction of introducing a data intelligence concept within the ShareTendPro network that can be used to predict the outcome of the tendering project.

8.3.4 Lack of political will

This can only be regarded as a shortcoming because most of the high-ranking positions within these organisations can be influenced by political will. Hence, the political will might be reluctant to adopt a solution that seeks to manage issues associated with monitoring the tendering projects since some of them might be linked to these projects.

8.4 Conclusion

In conclusion, Table 8.1 depicts a summary of the benefits and shortcomings that were identified by this research study.

Table 8.1 Summary of the benefits and shortcomings of this research study

No.	Primary benefits	Secondary benefits
1.	Distributed nature of the ShareTendPro network	Time efficiency
2.	Enhanced security	Cost reduction
3.	Greater transparency	Credible evidence
4.	Automation	Promotes real-time auditing and investigation
5.		Enforce trust
6.		Improve collaboration
Shortcomings		
1.	Storage constraints	
2.	off-chain storage	
3.	Scalability	
4.	Lack of political will	

This chapter has explored the benefits of this study including the shortcomings that might be encountered. It is evident that having a ShareTendPro network that might be used to securely share project information among various participants might eradicate issues related to the illegal altering of project information for corruption purposes and fostering accountability within the tendering system. However, this study has demonstrated that the benefits of the ShareTendPro network do not only reduce issues related to the illegal altering of project information, and securing project information, but also include the enforcement of trust among participants and provide better collaboration between law enforcement agents (auditors and investigators).

The following chapter provides a conclusion which sheds light on the contributions made by this study and confirmation of addressing the problem statement.

9. Chapter 9: Conclusion

9.1 Introduction

The previous chapter has evaluated this study using the following criteria, namely the benefits and shortcomings of the research. The benefits of the research criteria were classified into two categories namely primary and secondary benefits. The primary benefits focused on the benefits of the ShareTendPro network, while the secondary benefits focused on the benefits experienced by users as they interact with the ShareTendPro network. The shortcomings of the research criteria focused on issues that might affect the performance of the ShareTendPro network or issues that need to be considered when implementing the next version of the ShareTendPro network.

Therefore, this chapter provides a summary of this study while aiming to shed light on the contribution made by this research. The remainder of this chapter is structured as follows: a summary of all the chapters contained within this study is detailed. The recap of the problem statement is detailed in the following section. The highlights of the future work are presented in the section to follow, while the last section details the concluding remarks of the study.

9.2 Summary of all the chapters

This section summarises the discussions of all the chapters that are contained within this research study, with an aim of outlining the purpose of each chapter as part of trying to provide the reader with a better understanding of how the entire dissertation was structured.

- **Chapter 1** was an introductory chapter that sought to establish the problem statement of this study. Additionally, this chapter also explored the objectives and methodology used to address the identified problem, i.e., the fraud that could be committed due to using paperwork to share project information.
- **Chapter 2** provided the background literature on the tendering systems used by the South African Local Government (SALG).
- **Chapter 3** provided the background literature on the distributed ledger technology (DLT) used to implement the proposed solution. Additionally, this chapter explored different types of Blockchain technologies, including frameworks that might be used to implement the proposed solution. This chapter also explored the related work that can be associated with this research study.

- **Chapter 4** introduced the proposed model by outlining the role players or stakeholders of the current project information-sharing concept used by the SALG tendering system.
- **Chapter 5** explored the model design of the proposed solution that can be used to secretly share project information.
- **Chapter 6** depicted the implementation of the proposed solution, which focuses on the configuration details of the ShareTendPro model used to share project information. Note that the implementation of the proposed solution focused on a prototype that is aimed at demonstrating whether DLT might be the feasible technology solution used to secretly share project information.
- **Chapter 7** sought to demonstrate the results generated after executing the implementation of the proposed solution.
- **Chapter 8** provided the criteria used to evaluate this study by highlighting some of the benefits and shortcomings that were identified during this research study.
- Lastly, **Chapter 9**, provides the conclusion of this study by summarising the entire research. Additionally, this chapter also explores some of the ideas that should be considered in the future, for further research.

The following section revisits the problem statement highlighted in Chapter 1 to examine the extent to which the identified problem was achieved.

9.3 Recap of the problem statement

The primary problem this research study sought to address is the use of paperwork to share project information among various organisations since it might contribute towards illicit altering of project information during the process. This might also affect the fairness, transparency, data integrity, and competitiveness of the tendering system used by the SALG.

To address this problem, Chapter 1 highlighted the following research questions:

- **Research Question 1:** How does the tendering system work in the South African context? This question was answered in the following way. It was addressed by Chapter 2 because it explored the background literature on how the tendering system works. Chapter 2 also gave the context of how the organs of the state (i.e., Local Municipality or District Municipality) that fall within the SALG should adopt the tendering system.
- **Research Question 2:** What are the laws and principles that govern the procurement process? This question was answered in the following way. It was also addressed by

Chapter 2 because it explored the legislative frameworks and pillars of procurement that govern the procurement processes of the tendering system within the SALG.

- **Research Question 3:** Is DLT a possible solution to the identified problem? This question was answered in the following way. It was addressed by implementing the ShareTendPro network prototype that might be used to securely share project information among participants that have an interest in a tendering project. The ShareTendPro network prototype was implemented in Chapter 6 and the test results of it were discussed in Chapter 7.
- **Research Question 4:** How does transparency, accountability, and integrity of data or information in a potential solution work and how will it contribute to digital forensic investigations? This question was answered in the following way. The transparency, accountability, and data integrity part of this research question was addressed in Chapter 3, and these information security services are also highlighted in other chapters such as Chapters 6, 7, and 8. The issue of how the ShareTendPro network might contribute towards digital forensic investigation was highlighted in Chapters 7 and 8.

The following section provides the details that seek to highlight some areas for further research.

9.4 Future work

The implementation of the proposed solution, which is the ShareTendPro solution shows the potential of sharing tendering project information within the SALG. One of the main benefits is that the proposed solution seeks to enforce collaboration among various organisations that have an interest in tendering project information by providing real-time data. The access to real-time data also assists other organisations such as Auditor's Firm and Investigator's Firm to audit or investigate a specific tendering system in real-time, which also promotes accountability within that tendering project. However, during the research process, some new areas of interest that fall outside the scope of this study were discovered which can be considered for further research. Hence, the following items explore these areas of interest. Note that the last three bullets echo the further research as identified by the shortcomings of this research.

- The ShareTendPro network is based on the backend development, therefore, future research can focus on the implementation of a frontend application that would be used as the user interface of the ShareTendPro network. Additionally, the frontend application should be web-based to accommodate all users.

- To expand the scope of the ShareTendPro network by including other stakeholders that fall within the South African Government (SAG), or for any other government, which include all the three spheres namely Local, Provincial, and National Government.
- To address the storage constraints issues that were highlighted in the previous chapter, an implementation of a mechanism that will enable the ShareTendPro network to store project information using servers of various organisations that are part of the network would be considered for future purposes.
- To implement or configure the ShareTendPro network and allows it to store its information using servers of various organisations that are part of the network.
- To implement a mechanism that can be used by various computers to temporarily store project information when it becomes inactive and share that information with other nodes within the network when it becomes active.
- To deal with large volumes of data in future, then an implementation of a mechanism that might be used for data intelligence which would be required to analyse and transform massive datasets into intelligent data insights to improve service delivery or to fast track some of the processes. For example, the representation of the data stored within the ShareTendPro network using graphic visualisation will highlight some of the areas that need to be prioritised within the tendering project. Additionally, this mechanism will also speed up some of the processes related to the compilation of either auditor's or investigator's report since they would have access to the visual representation of the entire project history. Hence, the implementation of a data intelligence mechanism will help the participants of the ShareTendPro network to understand the project information stored within the network, which also promotes better decision-making within the tendering project.

The following section focuses on the final concluding remarks of the research study.

9.5 Conclusion

The ShareTendPro network demonstrates how Blockchain technology as a tool can be used to securely share project information with all parties that have an interest in the tendering project. The ShareTendPro network secures its project information using various information security mechanisms such as distributed ledger, cryptographic encryptions, timestamps, and having immutable data or transactions. The distributed ledger mechanism ensures that the tendering project information is readily available to all the participants of the ShareTendPro network.

The cryptographic encryptions ensure that the ShareTendPro network achieves confidentiality over project information. The timestamp mechanism ensures that the ShareTendPro network achieves data integrity over the tendering project information. The mechanism that seeks to ensure that the project information is immutable is also aimed at achieving data integrity within the ShareTendPro network.

One of the advantages of the ShareTendPro network is that it uses DLT, which is a technology that seeks to promote the need of sharing information while eliminating issues related to a single point of failure or having an organisation that has central powers over project information. Additionally, the use of DLT as an underlying technology also incorporates the benefits and promises that come with this new technology. Hence, this research study serves as the foundation for the next development of the next vision of the ShareTendPro network.

The SALG uses a tendering system to promote public and private partnerships, therefore, the ShareTendPro network becomes an essential platform for securely sharing project information without colluding. Additionally, the ShareTendPro network will provide greater transparency and accountability over project information, while enforcing trust on the other hand because no one will have to worry about the illegal altering of project information that might be processed unnoticed. This ensures that the ShareTendPro network stores credible digital evidence of the tendering project that can be used to depict the entire project history of a particular project of interest. Furthermore, the ShareTendPro network ensures that all the participants have instant access to real-time data stored within the Blockchain network without having to worry about issues that emanate from requesting that data directly from a specific organisation because some of them might be reluctant to share it.

The ShareTendPro network, therefore, would provide a revolutionary step towards curbing corruption in countries, like South Africa, where corruption currently enjoys high tide. It is hoped that other countries will also adopt this scheme.

References

- [1] P. Ramazhamba, T. Mashiane and Z. Dlamini, "Grade Monitoring System: A Prototype for Thulamela Secondary Schools," in *2018 IST-Africa Week Conference (IST-Africa)*, Gaborone, 2018.
- [2] N. Kshetri, "Will Blockchain emerge as a tool to break the poverty chain in the Global South?," *Third World Quarterly*, vol. 38 , no. 8, pp. 1710-1732, 2017.
- [3] Z. Arsovski, P. Lula and A. Dordevic, "IMPACT OF ICT ON QUALITY OF LIFE," in *Center for Quality*, 2016.
- [4] Department of Science and Technology (RSA), "ICT RDI Roadmap," [Online]. Available: <http://digitaladvantage.co.za/the-ict-rdi-roadmap/>. [Accessed 30 August 2018].
- [5] South African Revenue Service (SARS), "Home," SARS eFiling (Revenue Service), [Online]. Available: <https://secure.sarsefiling.co.za/!Generator/WebWiz.aspx?BusinessProcessCode=BPLlogin&ProcessCode=Login&ActionCode=Load>. [Accessed 28 September 2018].
- [6] Home Affairs, "Home," eHomeAffairs, [Online]. Available: <https://ehome.dha.gov.za/echannel>. [Accessed 10 09 2018].
- [7] S. Ngobeni, "An analysis of the tender process in national government in South Africa," in *MBA Thesis, North-West University, Potchefstroom Campus*, Potchefstroom , 2011.
- [8] P. Bolton, "Government procurement as a policy tool in South Africa," *Journal of Public Procurement* , vol. 6, no. 3, pp. 193-217, 2006.
- [9] C. Waters and D. Waters, "Operations management: producing goods and services," in *Pearson Education*, 2002.
- [10] A. Carstensen and J. Bernhard, "Design science research—a powerful tool for improving methods in engineering education research," *European Journal of Engineering Education*, vol. 44, no. 1-2, pp. 85-102, 2019.
- [11] YES!MEDIA, "The National Government Handbook South Africa," 2018. [Online]. Available: <http://www.yesmedia.co.za/the-national-government-handbook/>. [Accessed 29 11 2018].

-
- [12] National Government of South Africa, "Economic & Infrastructure Development," National Government, [Online]. Available: <https://nationalgovernment.co.za/units/view/212/council-for-scientific-and-industrial-research-csir>. [Accessed 29 11 2018].
- [13] South African Government, "About," Republic of South African , [Online]. Available: <https://www.gov.za/about-government/government-system/local-government>. [Accessed 02 10 2018].
- [14] Transparency International, "news," Transparency International, 21 February 2018. [Online]. Available: https://www.transparency.org/news/feature/corruption_perceptions_index_2017. [Accessed 2018 October 08].
- [15] Corruption Watch, "Learn About Corruption," News fighting corruption in South Africa, [Online]. Available: <https://www.corruptionwatch.org.za/wp-content/uploads/2018/04/Corruption-Watch-Annual-Report-04042018-FA-Single-Pages-CompressedV2-2.pdf>. [Accessed 26 September 2018].
- [16] Capricorn District Municipality, "Our Background," 23 June 2014. [Online]. Available: <https://www.cdm.org.za/about-us/our-background>. [Accessed 06 September 2018].
- [17] Republic of South Africa: National Treasury, "Legislation: PFMA-Supply Chain Management: General Procurement Guidelines," [Online]. Available: <http://www.treasury.gov.za/legislation/pfma/supplychain/General%20Procurement%20Guidelines.pdf>. [Accessed 13 12 2018].
- [18] P. Munzhedzi, "South African public sector procurement and corruption: Inseparable twins?," *Journal of Transport and Supply Chain Management*, vol. 10, no. 1, pp. 1-8, 2016.
- [19] Council of Supply Chain Management Professionals (CSCMP), "Certify," CSCMP Fundamentals, [Online]. Available: https://cscmp.org/CSCMP/Certify/Fundamentals/What_is_Supply_Chain_Management.aspx. [Accessed 06 11 2018].
- [20] K. Croxton, S. Garcia-Dastugue, D. Lambert and D. Rogers, "The supply chain management processes," *The International Journal of Logistics Management*, vol. 12, no. 2, pp. 13-36, 2001.

-
- [21] International Organization for Standardization (ISO), "ISO 10845-1:2010(en)," Construction procurement — Part 1: Processes, methods and procedures, 2010. [Online]. Available: <https://www.iso.org/obp/ui/#iso:std:iso:10845:-1:ed-1:v1:en>. [Accessed 07 11 2018].
- [22] D. Pooe, C. Mafini and D. Makhubele, "Investigating municipal procurement challenges in South Africa: a qualitative study," *The International Business & Economics Research Journal*, vol. 14, no. 1, pp. 67 - 78, 2015.
- [23] K. Wall, R. Watermeyer and G. Pirie, " " Wagging the dog": How service delivery can lose its way in the procurement maze--and could find it again," in *IMESA Conference 2012*, 2012.
- [24] M. Mathonsi and W. Thwala, "Factors influencing the selection of procurement systems in the South African construction industry.," *African Journal of Business Management*, vol. 6, no. 10, pp. 3583-3594, 2012.
- [25] R. RAMEEZDEEN and S. RATNASABAPATHY, "A multiple decisive factor model for construction procurement system selection," in *Conference Proceedings of the 6th Annual Research Conference, the Royal Institution of Chartered Surveyors*, College London, UK., 2006.
- [26] W. Thwala and M. Mathonsi, "Selection of procurement systems in the South African construction industry: An exploratory study," *Acta Commercii*, vol. 12, no. 1, pp. 13-26, 2012.
- [27] M. Saad, "Investigating public sector client performance in South African construction procurement," University of the Witwatersrand, Johannesburg., 2017.
- [28] R. Rashid, I. Taib, W. Ahmad, M. Nasid, W. Ali and Z. Zainordin, "Effect of procurement systems on the performance of construction projects," Universiti Teknologi Malaysia, 2006.
- [29] Department: National Treasury of the Republic of South Africa, "Practice Note SCM 1 of 2003: General Conditions of Contract (GCC) and Standardized Bidding Documents (SBDs)," 2003. [Online]. Available: <http://www.treasury.gov.za/divisions/ocpo/sc/PracticeNotes/default.aspx>. [Accessed 12 11 2018].

-
- [30] Department of Environmental Affairs of the Republic of South Africa, "Legislations: Accounting officers SCM guide," February 2004. [Online]. Available: https://www.environment.gov.za/sites/default/files/legislations/accountingofficer_sc_mguide.pdf. [Accessed 24 10 2018].
- [31] Republic of South Africa: NATIONAL TREASURY , "MFMA Guidelines: Supply Chain Management," October 2005. [Online]. Available: http://mfma.treasury.gov.za/MFMA/Guidelines/Guide%20for%20Municipal%20Accounting%20Officers_1.pdf. [Accessed 24 10 2018].
- [32] I. Ambe, "Public procurement trends and developments in South Africa," UNISA, 2016.
- [33] K. Moeti, T. Khalo, J. Mafunisa and S. Nsingo, "Public finance fundamentals," Juta, Cape Town, 2007.
- [34] K. Lysons and B. Farrington, "Purchasing and supply chain management," Pearson Education, 2006.
- [35] Republic of South Africa: Department of Trade and Industry, "Economic Empowerment," the dti, [Online]. Available: http://www.dti.gov.za/economic_empowerment/bee.jsp. [Accessed 15 April 2019].
- [36] News24, "WATCH LIVE: Mzwanele Manyi takes the stand at state capture inquiry," news24 Video, 14 November 2018. [Online]. Available: <https://www.youtube.com/watch?v=YHnmgsSjnjc>. [Accessed 12 12 2018].
- [37] Republic of South Africa: The Government Communication and Information System (GCIS), "Services," Government communications, [Online]. Available: <https://www.gcis.gov.za/content/services>. [Accessed 12 12 2018].
- [38] Republic of South Africa: South African Government, "Documents: The Constitution of the Republic of South Africa - Chapter 13: 213-230 Finance," Constitution, 1996. [Online]. Available: <https://www.gov.za/documents/constitution-republic-south-africa-1996>. [Accessed 13 12 2018].
- [39] Republic of South Africa: National Treasury, "Legislation: Local Government - Municipal Finance Management Act, No. 56 of 2003," 2003. [Online]. Available: <http://mfma.treasury.gov.za/Legislation/lgmfma/Pages/default.aspx>. [Accessed 13 12 2018].

-
- [40] Republic of South Africa: South African Government, "Documents: Preferential Procurement Policy Framework Act," 2000. [Online]. Available: <https://www.gov.za/documents/preferential-procurement-policy-framework-act>. [Accessed 13 12 2018].
- [41] Republic of South Africa: South African Government, "Documents - Acts: Public Finance Management Amendment Act," 1999. [Online]. Available: <https://www.gov.za/documents/public-finance-management-amendment-act>. [Accessed 14 12 2018].
- [42] F. Otieno, "The roles of monitoring and evaluation in projects," in *2nd International Conference on Construction in Developing Countries: Challenges facing the construction industry in developing countries*, 2000.
- [43] H. Venter and J. Eloff, "A taxonomy for information security technologies," *Computers & Security*, vol. 22, no. 4, pp. 299-307, 2003.
- [44] H. Venter and J. Eloff, "Network security: Important issues," *Network Security*, vol. 2000, no. 6, pp. 12-16, 2000.
- [45] G. Kessler, "An overview of cryptography," 30 April 2019. [Online]. Available: <https://www.garykessler.net/library/crypto.html#intro>. [Accessed 13 08 2019].
- [46] O. LaBarre, "Enterprise Resource Planning (ERP)," *Financial Analysis*, 14 04 2019. [Online]. Available: <https://www.investopedia.com/terms/e/erp.asp>. [Accessed 27 05 2019].
- [47] ASTRI, "Whitepaper on Distributed Ledger Technology," 11 November 2016. [Online]. Available: <https://www.astri.org/tdprojects/whitepaper-on-distributed-ledger-technology/>. [Accessed 18 02 2019].
- [48] H. Natarajan, S. Krause and H. Gradstein, "Distributed Ledger Technology and Blockchain," *The World Bank Group: Open knowledge Repository*, 2017.
- [49] Google, "Data and Security," *Google Data Centers*, [Online]. Available: <https://www.google.com/about/datacenters/inside/data-security/index.html>. [Accessed 08 July 2019].
- [50] C. Majaski, "Distributed Ledgers," *Investopedia*, 26 04 2019. [Online]. Available: <https://www.investopedia.com/terms/d/distributed-ledgers.asp>. [Accessed 28 05 2019].

-
- [51] N. Chalaemwongwan and W. Kurutach, "State of the art and challenges facing consensus protocols on blockchain," in *International Conference on Information Networking (ICOIN)* (pp. 957-962). IEEE, 2018.
- [52] S. Brakeville and B. Perepa, "Blockchain basics: Introduction to distributed ledgers," IBM, [Online]. Available: <https://developer.ibm.com/tutorials/cl-blockchain-basics-intro-bluemix-trs/>. [Accessed 13 June 2019].
- [53] M. Andoni, V. Robu, D. Flynn, S. Abram, D. Geach, D. Jenkins, P. McCallum and A. Peacock, "Blockchain technology in the energy sector: A systematic review of challenges and opportunities," in *Renewable and Sustainable Energy Reviews*, 100, pp.143-174, 2019.
- [54] bitcoin, "Mining," bitcoin wiki, [Online]. Available: <https://en.bitcoin.it/wiki/Mining>. [Accessed 30 05 2019].
- [55] C. Rajendran, "How Does Bitcoin Works in Detailed," LinkedIn, 2018 October 07. [Online]. Available: <https://www.linkedin.com/pulse/blockchain-development-part-2-understanding-bitcoin-rajendran/>. [Accessed 04 June 2019].
- [56] B. Forouzan, "Foundations of Computer Science, fourth edition," in *Algorithm representations*, 682, Cengage Learning (Emea) Ltd, 2017.
- [57] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," www.bitcoin.org, 2008.
- [58] W. Dai, "B-money," 1998. [Online]. Available: <http://www.weidai.com/bmoney.txt>. [Accessed 28 01 2019].
- [59] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84-90, 1981.
- [60] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *In Workshop on distributed cryptocurrencies and consensus ledgers*, 2016.
- [61] hyperledger-fabric, "Key Concepts: Ledger," [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-1.4/ledger/ledger.html>. [Accessed 09 October 2019].
- [62] Blockchain.com, "Explorer: BTC/Block," Blockchain, [Online]. Available: <https://www.blockchain.com/btc/block/600987>. [Accessed 25 10 2019].
- [63] K. Wüst and A. Gervais, "Do you need a Blockchain?," in *In 2018 Crypto Valley Conference on Blockchain Technology (CVCBT)* (pp. 45-54), 2018.

-
- [64] M. Vukolić, "HyperledgerFabric: An open-source distributed operating system for permissioned blockchainsSwiss," 22 June 2017. [Online]. Available: <https://blockchain-summer.epfl.ch/talks/hyperledger-fabric-vukolic.pdf>. [Accessed 29 03 2019].
- [65] R. Zagone, "Results of the Bank of England/Ripple Proof of Concept Published Today," Ripple | Insight, 10 July 2017. [Online]. Available: <https://ripple.com/insights/results-of-the-bank-of-englandripple-proof-of-concept-published-today/>. [Accessed 27 03 2019].
- [66] Ripple, "XRP the digital asset for payments," Ripple | XRP, [Online]. Available: <https://ripple.com/xrp/>. [Accessed 28 03 2019].
- [67] <https://xrpl.org>, "Docs: Introduction to Consensus," XRP Ledger, [Online]. Available: <https://xrpl.org/intro-to-consensus.htmlxrpl.prg>. [Accessed 25 11 2019].
- [68] M. Vukolić, "Hyperledger fabric: towards scalable blockchain for business. Tech. rep. Trust in Digital Life 2016," IBM Research, 17 June 2016. [Online]. Available: https://trustindigitallife.eu/wp-content/uploads/2016/07/marko_vukolic.pdf. [Accessed 29 03 2019].
- [69] D. Burkhardt, M. Werling and H. Lasi, "Distributed Ledger," in *2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, Stuttgart, Germany , 2018.
- [70] D. Schwartz, N. Youngs and A. Britto, "The ripple protocol consensus algorithm," Ripple Labs Inc White Paper 5(8), 2014.
- [71] P. Hegedus, "Towards Analyzing the Complexity Landscape of Solidity Based Ethereum Smart Contracts," in *2018 IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*, Gothenburg, Sweden, Sweden , 2018.
- [72] A. Joshi, M. Han and Y. Wang, "A survey on security and privacy issues of blockchain technology," *Mathematical Foundations of Computing*, vol. 1, no. 2, pp. 121-147, 2018.
- [73] Hyperledger, "About," Hyperledger Projects, [Online]. Available: <https://www.hyperledger.org/about>. [Accessed 19 03 2019].
- [74] A. Baliga, "Understanding blockchain consensus models," in *In Persistent*, 2017.

-
- [75] Hyperledger, "Hyperledger," Hyperledger Frameworks, [Online]. Available: <https://www.hyperledger.org/>. [Accessed 19 03 2019].
- [76] K. Olson, M. Bowman, J. Mitchell, S. Amundson, D. Middleton and C. Montgomery, "Sawtooth: An Introduction," in *The Linux Foundation*, 2018.
- [77] P. Sajana, M. Sindhu and M. Sethumadhavan, "On Blockchain Application: Hyperledger Fabric and Ethereum," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 18, pp. 2965-2970., 2018.
- [78] S. Rouhani and R. Deters, "Performance analysis of ethereum transactions in private blockchain," in *2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, Beijing, China , 2017.
- [79] D. Mali, D. Mogaveera, P. Kitawat and M. Jawwad, "Blockchain-based e-tendering system," *In 2020 4th International Conference on Intelligent Computing and Control Systems*, pp. 357-362, 2020.
- [80] X. Li, "BCES: A BlockChain based Credible E-Bidding System," in *IEEE 6th International Conference on Computer and Communications*, 2020.
- [81] T. Weingärtner, D. Batista, S. Köchli and G. Voutat, "Prototyping a Smart Contract Based Public Procurement to Fight Corruption.," *Computers*, vol. 10, no. 7, p. 85, 2021.
- [82] Y. Goswami, A. Agrawal and A. Bhatia, "E-Governance: A Tendering Framework Using Blockchain with Active Participation of Citizens," *In 2020 IEEE International Conference on Advanced Networks and Telecommunications Systems*, 12 2020.
- [83] Linux Foundation Projects, "Hyperledger-composer," Hyperledger Foundation, August 2021. [Online]. Available: <https://www.hyperledger.org/use/composer>. [Accessed 16 05 2022].
- [84] J. Deshpande, M. Gowda, M. Dixit, M. Khubbar, B. Jayasri and S. Lokesh, "Permissioned blockchain based public procurement system," *Journal of Physics: Conference Series*, vol. 1706, no. 1, 2020.
- [85] I. Omar, R. Jayaraman, M. Debe, K. Salah, I. Yaqoob and M. Omar, "Automating procurement contracts in the healthcare supply chain using blockchain smart contracts," in *EEE Access*, 9, 2021.

-
- [86] D. Čeke, N. Buzadija and S. Kunosić, "Enhancing transparency and fairness in public procurement process with the support of blockchain technology: a smart contract based approach," *21st International Symposium INFOTEH-JAHORINA*, March 2022.
- [87] F. Zbinden and G. Kondova, "Economic development in Mexico and the role of blockchain," *Advances in Economics and Business*, vol. 7, no. 1, pp. 55-64, 2019.
- [88] M. Das, X. Tao, Y. Liu and J. Cheng, "A blockchain-based integrated document management framework for construction applications," in *Automation in Construction*, 133, 104001, 2022.
- [89] V. Hassija, V. Chamola, D. Krishna, N. Kumar and M. Guizani, "A blockchain and edge-computing-based secure framework for government tender allocation," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2409-2418, 2020.
- [90] S. Perera, S. Nanayakkara, M. Rodrigo, S. Senaratne and R. Weinand, "Blockchain technology: Is it hype or real in the construction industry?," *Journal of Industrial Information Integration*, vol. 17, p. 100125, 2020.
- [91] F. Hardwick, R. Akram and K. Markantonakis, "Fair and Transparent Blockchain Based Tendering Framework - A Step Towards Open Governance," *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering*, 2018.
- [92] O. Ogunlela, O. Ojugbele and R. Tengeh, "Blockchain technology as a panacea for procurement corruption in digital era," *International Journal of Research in Business and Social Science*, vol. 10, no. 4, 2021.
- [93] S. Quamara and A. Singh, "SChain: towards the quest for redesigning supply-chain by augmenting Blockchain for end-to-end management," *International Journal of Information Technology*, pp. 1-12, 2022.
- [94] Commission of Inquiry into State Capture "The Zondo Commission", "Testimony of Mr X," The Commission of Inquiry into state capture, 18 February 2020. [Online]. Available: <https://sastatecapture.org.za/site/hearings/date/2020/2/18>. [Accessed 29 June 2020].
- [95] IBM, "Interacting with Hyperledger Composer through RESTful API," IBM Developers Recipes, 17 June 2019. [Online]. Available:

- <https://developer.ibm.com/recipes/tutorials/interacting-with-hyperledger-composer-through-restful-api/#:~:text=strongly%20recommended%20beforehand.-,How%20Hyperledger%20Composer%20REST%20Works,into%20an%20open%20API%20definition..> [Accessed 10 August 2020].
- [96] Hyperledger Composer, "Introduction," [Online]. Available: <https://hyperledger.github.io/composer/latest/introduction/introduction.html>. [Accessed 18 12 2019].
- [97] Hyperledger-fabric, "Glossary," Hyperledger-fabric release 1.4 documentation, [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-1.4/glossary.html#smart-contract>. [Accessed 09 09 2020].
- [98] Docker, "What is a Container?," Docker, [Online]. Available: <https://www.docker.com/resources/what-container>. [Accessed 09 09 2020].
- [99] Hyperledger-fabric, "Cryptogen Commands," Commands Reference, [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-1.1/commands/cryptogen-commands.html>. [Accessed 28 09 2020].
- [100] EyeWitness, "DA demands in-field training for SAPS officers in the Western Cape," EyeWitness, March 2022. [Online]. Available: <https://ewn.co.za/2022/03/04/da-demands-in-field-training-for-saps-officers-in-the-western-cape>. [Accessed 28 March 2022].
- [101] South African Government, "MEC Albert Fritz on e-docket software not being effectively used by SAPS and Courts," Government speeches, 5 March 2020. [Online]. Available: <https://www.gov.za/speeches/r6135-million-e-docket-software-not-being-effectively-used-saps-courts-%C2%A0-5-mar-2020-0000>. [Accessed 28 March 2022].
- [102] H. Shacham and B. Waters, "Compact proofs of retrievability," in *International Conference on the Theory and Application of Cryptology and Information Security*, Berlin, Heidelberg, 2008.
- [103] IOTA, "What is IOTA," Distributed Ledger Technology, [Online]. Available: <https://www.iota.org/get-started/what-is-iota>. [Accessed 01 03 2019].
- [104] K. Croman, C. Decker, I. Eyal, A. Gencer, A. Juels, A. Kosba, A. Miller, E. Saxena, E. Shi, D. Sirer, D. Song and R. Wattenhofer, "n scaling decentralized blockchains,"

- in *International Conference on Financial Cryptography and Data Security*, Berlin, 2016.
- [105] S. Popov, "The Tangle," IOTA, 30 April 2018. [Online]. Available: <https://www.iota.org/research/academic-papers>. [Accessed 05 03 2019].
- [106] B. Kusmierz, "The first glance at the simulation of the Tangle: discrete model," IOTA, 06 11 2017. [Online]. Available: <https://www.iota.org/research/academic-papers>. [Accessed 01 03 2019].
- [107] Q. Bramas, "The Security and Stability of the Tangle," IOTA, 2018. [Online]. Available: <https://www.iota.org/research/academic-papers>. [Accessed 05 03 2019].
- [108] S. Philip, "Quasi-Analytic Parasite Chain Absortion Probabilities in the Tangle," IOTA, 31 12 2017. [Online]. Available: <https://www.iota.org/research/academic-papers>. [Accessed 05 03 2019].
- [109] IOTA, "Equilibria in the Tangle," IOTA-The Tangle algorithm, 03 03 2018. [Online]. Available: <https://www.iota.org/research/academic-papers>.
- [110] Y. Sompolinsky and A. Zohar, "Accelerating Bitcoin's Transaction Processing. Fast Money Grows on Trees, Not Chains," in *LACR Cryptology ePrint Archive*, 2013.
- [111] L. Baird, "The swirls hashgraph consensus algorithm: Fair, fast, byzantine fault tolerance," SWIRLDS TECH REPORT SWIRLDS-TR-2016-01, 2016.
- [112] M. Choe, "Open HashGraph: An Ultimate Blockchain Engine," 2018.
- [113] M. Ben-Or, "Another advantage of free choice (extended abstract): Completely asynchronous agreement protocols," in *Proceedings of the second annual ACM symposium on Principles of distributed computing*, 1983.
- [114] Hedera, "How it works," www.hedera.com, [Online]. Available: <https://www.hedera.com/how-it-works>. [Accessed 08 November 2019].
- [115] VirtualBox, "Welcome page," VirtualBox, [Online]. Available: <https://www.virtualbox.org/>. [Accessed 20 October 2020].
- [116] VirtualBox, "Status: Guest OSes," Guest_OSes, [Online]. Available: https://www.virtualbox.org/wiki/Guest_OSes. [Accessed 20 October 2020].
- [117] curl, "curl," open source software, [Online]. Available: <https://curl.haxx.se/>. [Accessed 01 11 2020].

-
- [118] Git, "Git --everything-is-local," Git open source version control system, [Online]. Available: <https://git-scm.com/>. [Accessed 01 11 2020].
- [119] Golang, "GO," Google open source software, [Online]. Available: <https://golang.org/>. [Accessed 01 11 2020].
- [120] Hyperledger-fabric, "Contributions welcome: Coding guidelines," Hyperledger-fabric, [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-1.3/Style-guides/go-style.html>. [Accessed 01 11 2020].
- [121] Opensource.com, "What is Python?," Opensource, [Online]. Available: <https://opensource.com/resources/python>. [Accessed 02 11 2020].
- [122] Python, "About/Apps," Python Web and Internet Development, [Online]. Available: <https://www.python.org/about/apps/>. [Accessed 03 11 2020].
- [123] yaml.org, "YAML 1.2," YAML, [Online]. Available: <https://yaml.org/>. [Accessed 09 03 2021].
- [124] Node.js, "About Node.js," Node.js, [Online]. Available: <https://nodejs.org/en/about/>. [Accessed 02 11 2020].
- [125] OpenSource, "What is Docker?," Open source software, [Online]. Available: <https://opensource.com/resources/what-docker>. [Accessed 01 11 2020].
- [126] Hyperledger-fabric, "getting Started - install Sample, Binaries, and Docker Images," Hyperledger Fabric, [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/latest/install.html?highlight=binaries>. [Accessed 03 11 2020].
- [127] N. Alpern and R. Shimonski, "Eleventh Hour Network+ : CHAPTER 1 - Network Fundamentals," in *Exam N10-004 Study Guide*, ScienceDirect, 2010, pp. 1 - 18.
- [128] IBM, "IBM Cloud Learn Hub / What is etcd?," IBM Cloud, 18 December 2019. [Online]. Available: <https://www.ibm.com/cloud/learn/etcd>. [Accessed 3 May 2021].
- [129] ETCD, "What is etcd?," ETCD, [Online]. Available: <https://etcd.io/>. [Accessed 03 May 2021].
- [130] RAFT, "The Raft Consensus Algorithm," RAFT, [Online]. Available: <https://raft.github.io/>. [Accessed 03 May 2021].
- [131] IBM, "Apache CouchDB," IBM Cloud, [Online]. Available: <https://www.ibm.com/cloud/learn/couchdb>. [Accessed 12 01 2021].

- [132] gobyexample.com, "Go by Example: Structs," gobyexample.com, [Online]. Available: <https://gobyexample.com/structs>. [Accessed 01 June 2021].
- [133] Go, "Go Shim," Go.dev documentation, [Online]. Available: <https://pkg.go.dev/github.com/3100/fabric/core/chaincode/shim>. [Accessed 01 June 2021].
- [134] gobyexample, "Go by Example: Slices," gobyexample.com, [Online]. Available: <https://gobyexample.com/slices>. [Accessed 01 June 2021].
- [135] "Consensus Algorithms," Medium , 21 September 2018. [Online]. Available: <https://medium.com/coinbundle/consensus-algorithms-dfa4f355259d>. [Accessed 14 May 2019].

A. Appendix A: Directed Acyclic Graph

Figure A.1 depicts one of the graphs, usually studied in Mathematics as Graph Theory and in Computer Science as Finite Automata. This structure consists of objects called nodes, represented by alphabets A to F which are linked using edges and each edge represents the relationship between these nodes. There are various ways of representing these relationships. In Mathematics a line with an arrow represents a directed relationship, while a solid line without arrows represents a bidirectional relationship. Additionally, in the field of Computer Science, a directed relationship represents how the system transitions from one state to the other. Figure A.1 represents a directed relationship with no cycles and this graph is also referred to as Directed Acyclic Graph (DAG) in Mathematics.

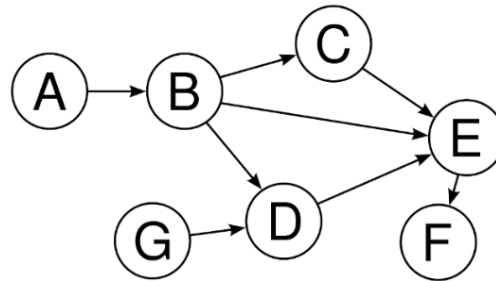


Figure A.1 Representation of the DAG [102]

There are various DLTs that use DAG as their data structure and each of these technologies represents data in a different way. However, this section focuses on the two famous DLTs that use DAG and these technologies are IOTA and Hashgraph. Therefore, the following sections explore how these technologies use DAG as their data structure.

A.1 IOTA

IOTA is an open-source, permission-less distributed ledger technology (DLT) designed with an intention of powering the Internet of Things (IoT) [103]. This technology primarily addresses the issue of scalability and transactions speed that seem to plague the existing BCT technologies [104]. Figure A.2 illustrates an IOTA data structure.

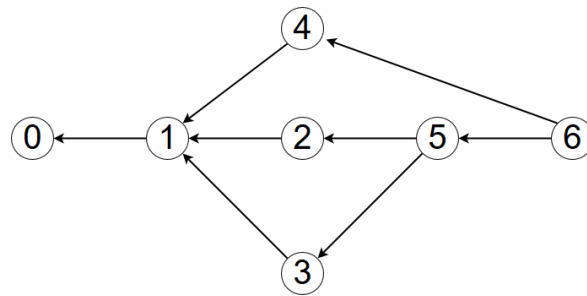


Figure A.2 IOTA (Tangle) illustration [54]

IOTA uses Tangle as its consensus algorithm and this algorithm uses DAG data structure to store its transactions [105]. Tangle does not group its transactions into blocks as presented by Blockchain data structure, however, its transactions are stored as a stream of individual transactions entangled together [103]. The process of adding new transactions to the network starts when a node digitally signs a transaction using its private-key. The mechanism of providing a digital signature also contributes towards data integrity and accountability since ownership of the transactions can be traced back to a specific node. This process requires each node to select two leaf nodes (also known as tips) that can be used to validate its transactions as shown in Figure A.2, except for the first two nodes (node 0 & node 1). Tips are all the nodes that have no entering edges and labelled 6 of Figure A.2 reflect such nodes [106]. IOTA makes use of an algorithm called Markov Chain Monte Carlo to select two trustworthy tips to validate its transaction [107]. Thereafter, the network will use Proof-of-Work (POW) algorithm to accept these transactions. This process continues and increases in speed as the network grows [108]. However, during the early stages of the network, IOTA uses coordinators to validate the transactions and this mechanism seeks to reduce the risks of having a Sybil attack due to the length of the chain [105].

Each node in IOTA has a weight and a cumulative weight. The weight represents how much work a specific node placed into the POW algorithm and the cumulative weight is the summation of all weights contributed by that node [109]. This cumulative weight acts as an indicator of whether a node is trustworthy or not. Hence, having the highest cumulative weight than others implies that it is a trusted node. IOTA also uses the Ghost protocol, proposed by Sompolinsky and Zohar in 2013 [105]. The Ghost protocol is regarded as the modification of the Bitcoin protocol by changing the main ledger of the Bitcoin system from Blockchain to DAG [110].

A.2 Hashgraph

Hashgraph is one of the emerging DLT data structures that uses DAG. Leemon Baird [111] invented Hashgraph in 2016, after publishing a white paper that explains an alternative way of using Byzantine Fault Tolerance protocol to achieve consensus. The Hashgraph achieves its consensus by using the virtual voting concept and gossip protocol [112]. The virtual voting concept, in this case, is the mechanism used by a particular node to calculate the ordering of transactions without the need of requesting votes from other nodes. As illustrated earlier on that gossip protocol is used to share new transactions with neighbouring nodes and it also shares the gossip that was obtained from other nodes. Hashgraph resolves the issue of asynchronous byzantine fault tolerance (aBFT) by applying the Ben-Or's algorithm [113] and gossip protocol [112]. A protocol that is based on Ben-Or's algorithm is regarded as fast, fair, cost-effective, and more secure compared to other DLT protocols since it consists of mathematical proof [112]. Figure A.3 represents a Hashgraph data structure.

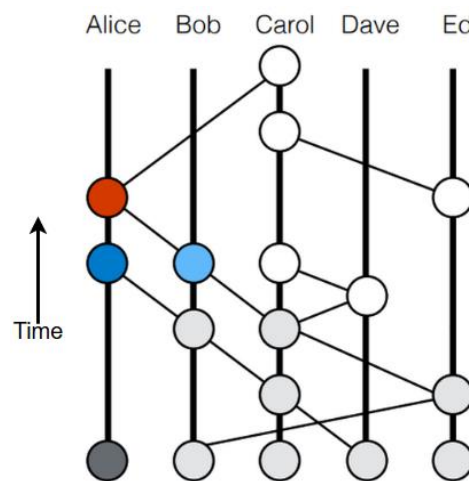


Figure A.3 Representation of the Hashgraph [112]

Hashgraph supports private or permissioned platforms as explained by Leemon Baird white paper, but it is owned by a company called Swirlds. This is against Satoshi's ideology of not relying on or having a trusted third party to govern the network. However, to overcome this issue Leemon and Hermon [114] introduced Hedera as an official public distributed ledger authorised to use the hashgraph algorithm. Additionally, no licence is required to use the Hedera API, but all the applications that run the Hedera API are required to make micropayments for using the platform tokens.

B. Appendix B: Detailed implementation of the ShareTendPro prototype

B.1 Introduction

Chapter 4 of this study proposed the ShareTendPro model that seeks to address the identified problem of using paperwork to share tendering project information, while Chapter 5 explored the detailed design of the proposed solution, which is called the ShareTendPro model. Chapter 5 further discussed the ShareTendPro model as well as the components associated with it, which were fully explained.

This chapter is devoted to the implementation of the ShareTendPro model as a prototype or proof of concept. Note that this entire chapter does not show the execution of the prototype (as purported by the model design of Chapter 5), but strictly its implementation details. Chapter 7 shows the running of the prototype in detail. Therefore, the remainder of this Appendix is structured as follows: the set of tools used to implement the prototype are discussed in detail, which include hardware and software specifications. The ShareTendPro model implementation is discussed in detail in the next section to follow. Additionally, this section explores the implementation of the ShareTendPro network topology and the chaincode that governs the ShareTendPro network topology. The last section provides a conclusion by summarising the chapter.

B.2 Tools used to develop the prototype

This section discusses the tools used to implement the proposed solution, which is the ShareTendPro prototype. This study has classified all the required tools into two categories namely hardware and software requirements. Therefore, the following sections explore these requirements in detail.

B.2.1 Hardware requirements

The hardware requirements can be viewed as a set of requirements defined by any operating system. However, this study classifies the hardware requirements into two categories namely physical and virtual machines. The physical machine can be viewed as a hardware-based device or computer, while the virtual machine is a software computer that seeks to emulate an actual physical machine. Additionally, this study makes use of VirtualBox to implement the desired

virtual machine used to develop the proposed solution. VirtualBox is regarded as a freeware tool that can be used to virtualise the X86 hardware or computing architecture [115]. Additionally, VirtualBox act as a hypervisor that creates a virtual machine that can be used by end-users to run another operating system (OS). A hypervisor can be viewed as firmware or software that manages the virtual machines. The operating system where VirtualBox runs on is referred to as the “host” OS, while the operating system running in the virtual machine is called the “guest” OS. Therefore, they are several guest OSs that are supported by the VirtualBox [115]. However, this study focuses on one of the guest OSs that falls within the Linux family, which is Ubuntu OS [116].

Table B.1 depicts all the requirements of both physical and virtual machines as used by this study.

Table B.1 Hardware requirements

Requirements	Physical Machine	Virtual Machine
Operating System	Windows 10	Ubuntu 18.4 (64-bit)
Base Memory (RAM)	16GB	8192MB ~ 8.2 GB (minimum required is 4GB)
Processor	Intel® Core (TM) i7-5600U CPU @ 2.60GHz 2.59GHz	2
Storage (Hard Disk Drive)	500GB	60GB
Network Adapter	Intel PRO/1000 MT Desktop, Intel (R) Dual Band Wireless-AC 7265	Intel PRO/1000 MT Desktop (Bridged Adapter)

Having introduced the Hardware requirements to the reader, then the next section presents the software requirements used to implement the proposed solution or prototype.

B.2.2 Software requirements

The software requirements explore the software tools required to implement the proposed solution or prototype. As indicated in Chapter 3, this study adopts the use of Hyperledger-fabric

(HLF) as a chosen Blockchain framework that might be used to implement the proposed solution. Therefore, this section explores the software tools required to run the HLF framework on a virtual machine. In other words, this section focuses on the prerequisites of the HLF framework. This section classifies the software used by this study into two categories namely development software and ShareTendPro software. The development software refers to all the software required to install the chosen Blockchain framework, while the ShareTendPro software is the software required to implement the proposed solution, i.e. ShareTendPro. Note that some software (for example, Go, Python, etc.) is used for both the development software as well as the ShareTendPro software (i.e. tick marks are found in both columns in Table B.2). The distinction between the use of the software in both cases is explained in the table.

Table B.2 depicts the software tools installed within the virtual machine. These software tools are all open-source tools, and they are highlighted in Table B.2 are explained in detail below.

Table B.2 Software

Software		Installed version	Development Software	ShareTendPro Software
cURL		7.58.0	✓	
Git		2.17.1	✓	
Go		1.15.2	✓	✓
Python		2.7.17	✓	✓
Node.js		10.23.0	✓	✓
Docker	Docker engine	19.03.6	✓	✓
	Docker-compose	1.17.1	✓	✓
HLF binaries, and docker images		2.0.1 1.4.6 (Fabric 2.0.1 & Fabric-CA 1.4.6)	✓	✓

The following items seek to explore the software tools installed within the virtual machine. For each of the software tools, first, a general description is given for the tool, followed by how the tool is used as development software and/or ShareTendPro software.

- **cURL**
 - *General description:* cURL is a software command-line-based tool used for transferring data including files using URL syntax [117]. In other words, cURL can be viewed as a command-line tool that seeks to upload or download data to and from a specific server, i.e., Github server.
 - *Use of development software:* cURL is used to download data of some of the software tools required to install the chosen Blockchain framework successfully, which is the HLF framework. For instance, a bootstrap script that would be discussed later requires the cURL tool to download a specific version of the HLF samples, binaries, and docker images. The HLF samples and binaries are available for downloaded from Github (at *github.com*), while the docker images are available for downloaded from docker (at *hub.docker.com*).
 - *Use for ShareTendPro software:* Not applicable.
- **Git**
 - *General description:* Git is a distributed version control system designed for tracking the changes in a source code during software development [118].
 - *Use for development software:* The chosen Blockchain framework contains several versions available, and these various versions of the framework are stored on Github. Hence, the Git tool is used for downloading a specific version of the HLF framework.
 - *Use for ShareTendPro software:* Not applicable
- **Go (also known as Golang)**
 - *General description:* Go is a programming language that makes it easy to build reliable and efficient software [119].
 - *Use for development software:* The HLF framework uses the Go tool for package management [120]. Hence, the Go tool can be viewed as a mechanism that handles the communication within the HLF framework. For instance, the HLF framework use Go tool to implement the mechanisms that enable various participants or organisations to secretly share project information with each other.
 - *Use for ShareTendPro software:* In the ShareTendPro model, the Go tool is used to implement the chaincode (smart-contract) that contains the rules that govern the virtual network of the proposed solution.

- **Python**

- *General description:* Python is one of the most popular programming languages that can be used for a wide variety of applications. These applications include a high-level data structure, dynamic typing, dynamic binding, and other features that are useful for complex application development as it is for scripting that connects components together [121].
- *Use for development software:* HLF framework relies on certain Python libraries (for example, YAML, JSON, etc.) for various purposes including virtual network configurations, documentations, and data structures. For instance, HLF framework uses a YAML (YAML is a recursive acronym for ‘YAML Ain’t Markup Language’) library for virtual network configurations, while the JSON (JavaScript Object Notation) library is used for data structures [122]. YAML can be viewed as a human-readable data serialisation language [123].
- *Use for ShareTendPro software:* the ShareTendPro model makes use of a YAML library for configuring all the necessary information required to implement the desired virtual network of the proposed solution.

- **Node.js**

- *General description:* Node.js is an asynchronous event-driven JavaScript runtime, which is designed to build scalable network applications [124].
- *Use for development software:* HLF framework makes use of the Node.js tool to implement and package all the necessary information required to run a virtual node.
- *Use for ShareTendPro software:* the ShareTendPro model collects some of the YAML configuration details and executes them to generate virtual nodes of the proposed solution. However, the configuration of these virtual nodes is discussed in detail later within the docker-compose section.

- **Docker**

- *General description:* Docker is a tool that can be used to create, deploy, and run applications by using containers in a virtualised fashion. Containers are like compact virtual machines (or nodes) that allow a developer to package an application with all the parts it needs (such as libraries, operating system, and other dependencies), and deploy it as one package [125].

- *Use for development software:* HLF framework makes use of the docker tools (docker-compose and docker-engine) to package all the necessary information required to run the virtual nodes of the proposed solution. Hence, these docker tools can be viewed as a mechanism that seeks to provide certain services required by the virtual network to distribute project information among various participants.
- *Use for ShareTendPro software:* In the ShareTendPro model, the Docker-compose tool is used to configure all the necessary information required to run a specific service, while the docker tool is used to package all the necessary information required to run a specific service into a single package or docker-container.
- **HLF binaries, and docker images**
 - *General description:* a cURL bootstrap script is used to download and install a specific version of the HLF framework's binaries (also known as binary files) and docker images. A binary file is a file that is written in a manner that can be read or understood by either a program or a hardware processor. In other words, binary files are computer-readable (non-text) files, but not human-readable (text) files. Some of the specific binaries involved are named *cryptogen*, *configtxgen* and *peer*. These binaries are discussed in the context that they are used, in more detail later. However, the pre-requisites for installing the HLF framework successfully requires that are all the above-mentioned software tools (cURL, Git, Go, Python, Node.js, Docker engine, and Docker-compose) are already installed.
 - *Use for development software:* the cURL bootstrap script is executed from the command prompt to download and install a specific version of the HLF binaries and docker images [126].
 - *Use for ShareTendPro software:* the ShareTendPro model makes use of the HLF binaries and docker images downloaded by the cURL bootstrap script to configure some of the necessary details required to execute the desirable virtual network of the proposed solution.

Figure B.1 depicts a visual representation of all the hardware and software requirements.

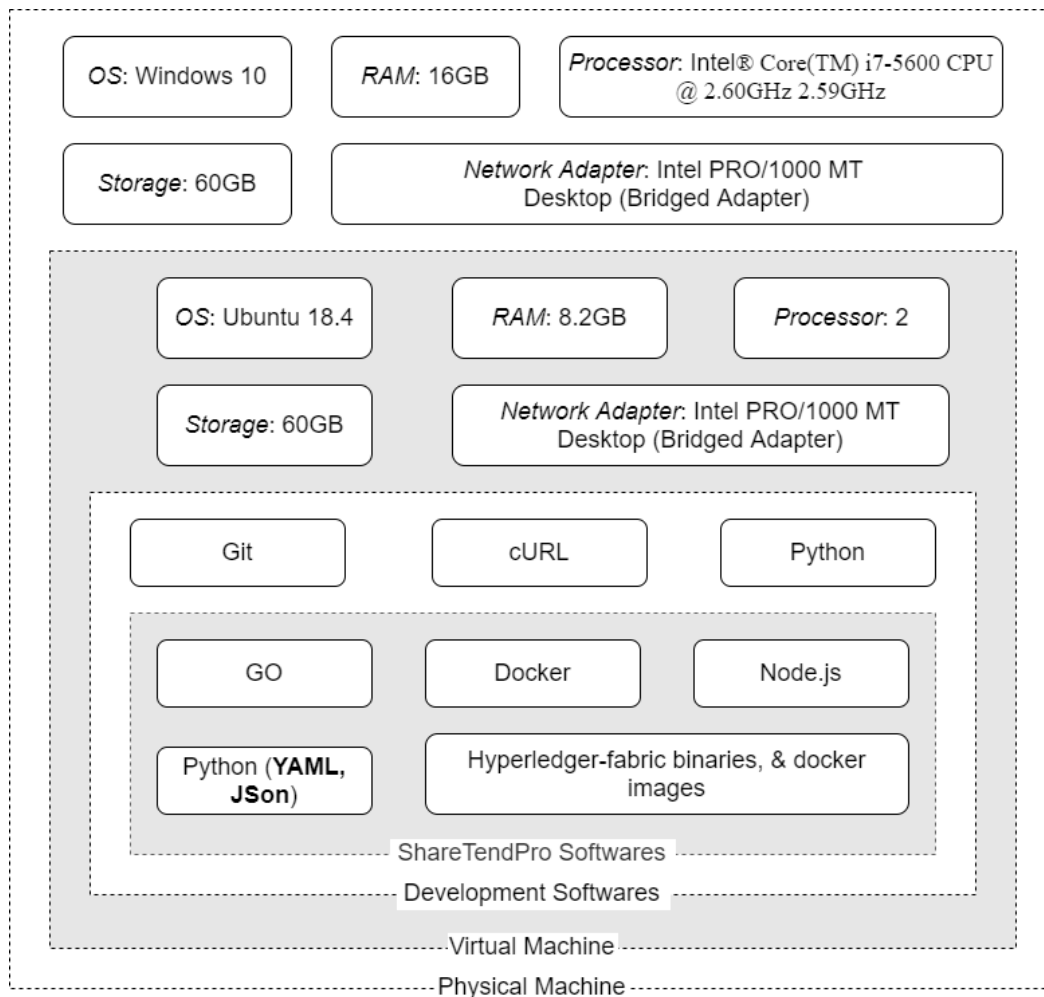


Figure B.1 Visual representation of hardware and software

All the necessary requirement details were explored within this section. However, it is mandatory to ensure that all the Software requirements are met since they are aimed at ensuring that the chosen Blockchain framework, which is HLF, runs or work successfully. Therefore, the following section delves into the implementation of the ShareTendPro model to accomplish this.

B.3 ShareTendPro model implementation

This section represents a detailed implementation of the ShareTendPro model, starting with a general overview of the ShareTendPro development process, followed by a more detailed discussion later. However, this section classified the model implementation (which is the ShareTendPro development process) into two high-level categories namely: *ShareTendPro network topology* and *chaincode development* as shown in Figure B.2. The correlation of how these two categories are related is highlighted later during the visualisation of the

ShareTendPro network topology in Figure B.5. The ShareTendPro network topology focuses on the configuration details related to the virtual network, while the chaincode development focuses on implementing the mechanism or rules that govern the virtual network. The ShareTendPro network topology category is classified into three subsections namely: crypto-config, configtx, and docker-compose as shown in Figure B.2. These subsections explore all the configuration details needed for implementing the desired ShareTendPro network topology (which is explained later in detail). A more detailed discussion of the ShareTendPro development process follows next.

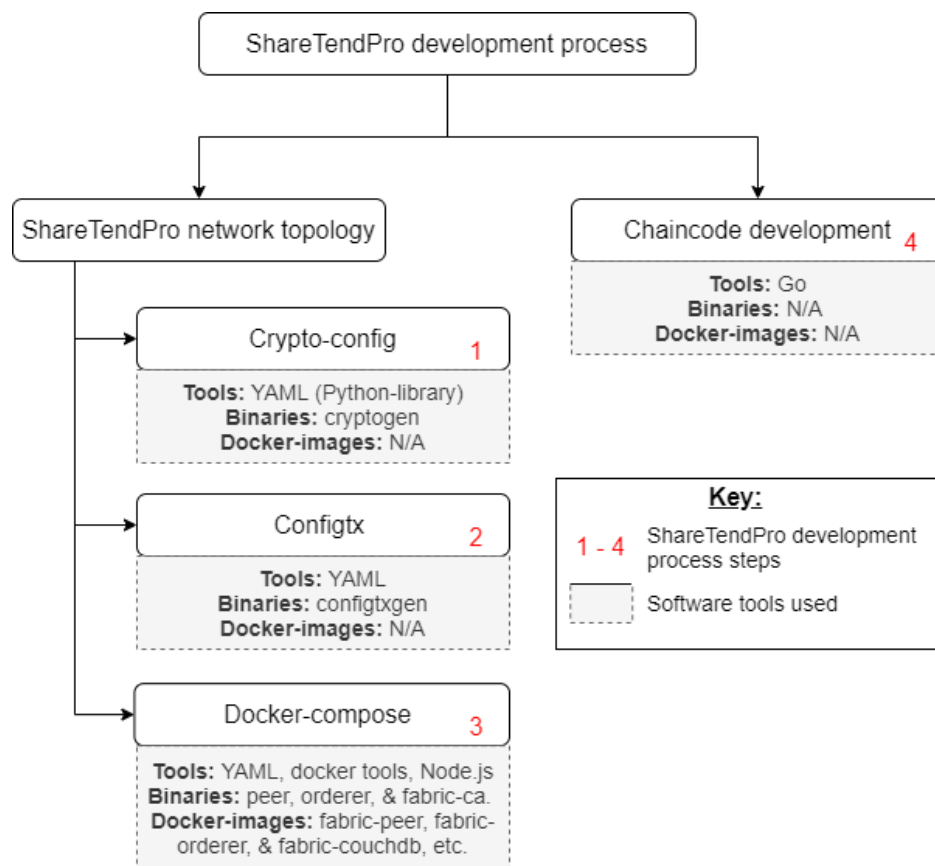


Figure B.2 ShareTendPro development process high-level

Figure B.2 depicts the adopted hierarchy of the ShareTendPro development process. Note that the process order in which the model was developed, is shown by the numbers 1 to 4, as indicated by the key in the figure. The configuration details related to the crypto-config category are compiled using the YAML library and fed to a binary file called *cryptogen* to generate the crypto materials that would be used to secure the communication within the proposed solution. The configuration details related to the configtx category are compiled using the YAML library and fed to the *configtxgen* binary file to generate channel artifacts. Channel artifacts can be viewed as the materials used by the HLF framework to generate a private

channel or virtual network. The configuration details related to the docker-compose category are compiled using the YAML library and fed to the docker-compose tool. The docker-compose tool then makes use of the docker-engine tool to generate the virtual nodes required by the proposed solution. Thereafter, the Node.js tool would use these virtual nodes to create a virtual network or a Blockchain network of the proposed solution. However, a specific virtual node requires certain binaries and docker images for it to be regarded as a successful virtual node of the Blockchain network.

The chaincode is compiled using the Go tool and it contains functions that allows various nodes within the proposed solution to interact with the virtual network of the proposed solution. Hence, all the functions within the chaincode development can be viewed as the rules that govern the virtual network of the proposed solution.

The details of the implementation of the ShareTendPro network topology, as well as the chaincode implementation, follow in the next two sections.

B.3.1 ShareTendPro network topology

This section explores all the necessary information required to generate a virtual network or Blockchain network of the proposed solution. As indicated earlier on, this section is divided into three categories namely crypto-config, configtx, and docker-compose and all these categories correspond with the three important files discussed in the previous chapter. However, the ShareTendPro network consists of other script files that might be used to either package some of the command lines required to run the network or generate the network artifacts (which are cryptographic materials, and these artifacts are explored in detail later). Script files can be viewed as a text document that contains instructions written in a specific scripting language, hence, these files are both human and machine-readable.

Figure B.3 depicts a more detailed hierarchy of the ShareTendPro network topology development process, compared to Figure B.2. Figure B.3 is explored in detail later.

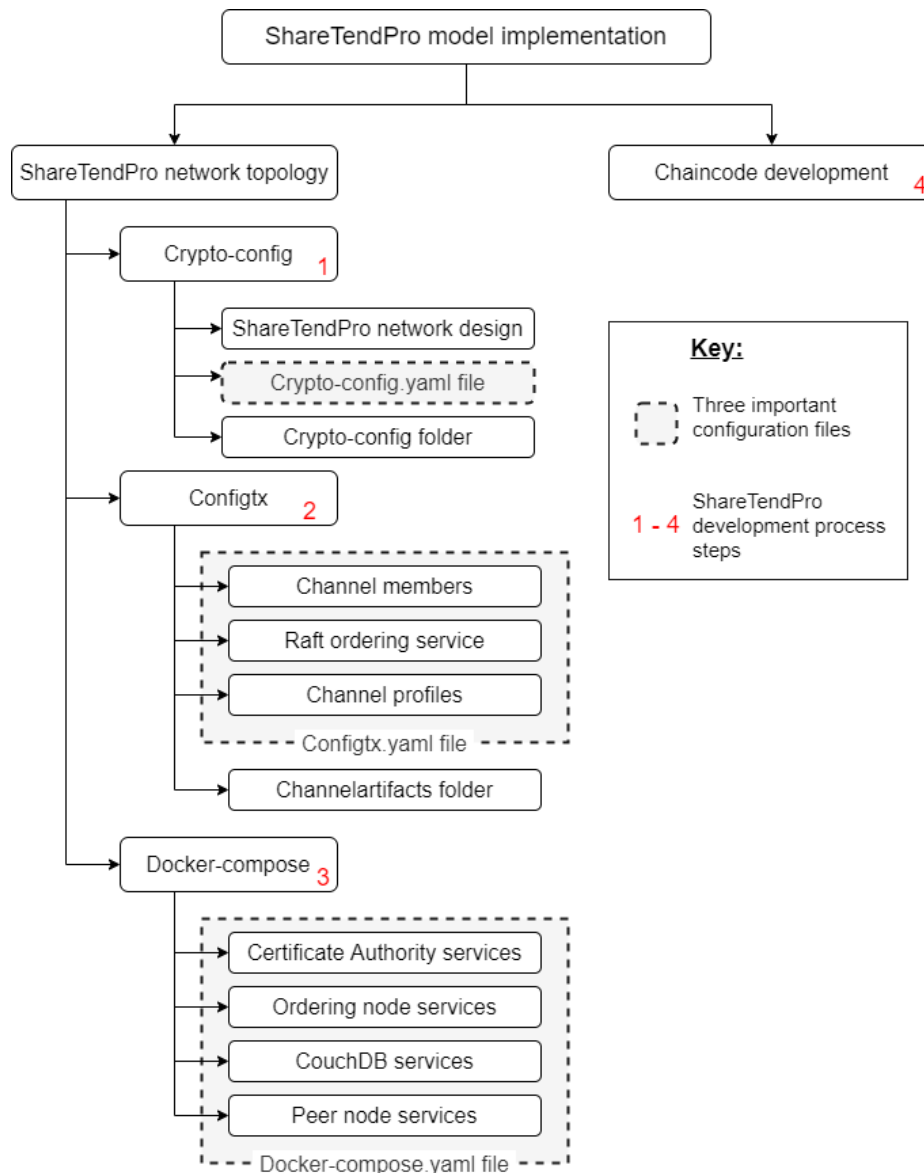


Figure B.3 ShareTendPro network topology development process

The following items seek to explore how these categories (i.e. crypto-config, configtx, and docker-compose) are implemented by the three important files (i.e. crypto-config.yaml, configtx.yaml, and docker-compose.yaml discussed in Figure 5.14):

- **Crypto-config:** contains all the configuration details related to the crypto-config.yaml file. However, the crypto-config category is divided into three sections namely:
 - *ShareTendPro network design* – this section seeks to visualise the desirable virtual network of the proposed solution. In other words, this section seeks to visualise the Blockchain network structure of the proposed solution, including how resources from various organisations might be utilised.

- *Crypto-config.yaml file* – this section seeks to explore the configuration details of all the participants or organisations and their resources (nodes and users). For instance, the resources of a Local Municipality are its members and the nodes that belong to that organisation.
- *Crypto-config folder* – this section seeks to examine the cryptographic materials generated after executing the *crypto-config.yaml* file. As indicated in the previous chapter, cryptographic materials (crypto materials) are nothing but digital certificates and keys that are used by a specific organisation and their resources to secure the communication channel as they interact with the virtual network or Blockchain network. However, the *crypto-config.yaml* file is fed into a binary file called *cryptogen* in order to generate the crypto materials of all the participants or organisations.
- **Configtx:** depicts all the configuration details related to the *configtx.yaml* file. The *configtx* category is divided into four sections namely: channel members, Raft ordering service, channel profiles, and channel-artifacts folder as shown in Figure B.3. However, the *configtx.yaml* file is composed of the first three sections (channel members, Raft ordering service, and channel profiles as seen in Figure B.3) due to the extensive information contained within the file. Therefore, the following items provide an overview of all four sections:
 - *Channel members* – this section contains the configuration details related to all the members of the communication channel, for example, Local Municipality, District Municipality, etc.
 - *Raft ordering service* – this section contains the configuration details of the ordering service used by the proposed solution. As presented in the previous chapter, ordering services consist of the ordering nodes that seek to add a new block of transactions to the ShareTendPro network.
 - *Channel profiles* – this section contains the configuration details related to the profiles that seek to generate the channel-artifacts. In other words, this section explores the profiles that generate a private communication channel that would be used by the channel members to secretly share project information.
 - *Channel-artifacts folder* – this section examines the results generated after executing the channel profiles within the *configtx.yaml* file and these results are stored within the channel-artifacts folder. However, the *configtx.yaml* file is fed

to a binary file called *configtxgen* in order to generate the channel artifacts or execute the channel profiles.

- **Docker-compose:** contains all the configuration details related to the *docker-compose.yaml* file. The *docker-compose* category is divided into seven sections namely: certificate authority services, ordering services, couchDB services, Local Municipality services, District Municipality services, Investigator’s services, and Auditor’s services. However, the *docker-compose.yaml* file is composed of all the seven sections as shown in Figure B.3. Additionally, this file is fed to a binary script called *peer* to generate all the necessary information required to create, join and update the channel communication. Therefore, the following items provide an overview of these sections:
 - *Certificate authority services* – this section explore all the configuration details related to all the services that seek to identify the resources that belong to a specific organisation within the Blockchain network.
 - *Ordering node services* – this section depicts the configuration details related to all the services that seek to add a new block of transactions to the ShareTendPro network.
 - *CouchDB services* – this section explores all the configuration details that seek to store the state of a Blockchain network. In a distributed ledger system, the couchDB services can be viewed as the world-state of the proposed solution.
 - *Peer node services* – this section explores all the configuration details of the peer nodes that belong to various organisations, such as Local Municipality, District Municipality, Investigator’s Firm, and Auditor’s Firm.

As indicated in Figure B.3, all these configurations correspond with the three important files (*crypto-config.yaml* file, *configtx.yaml* file, and *docker-compose.yaml* file) as discussed in the previous chapter (Figure 5.14). Note that all these configuration files are fed to some of the applications mentioned in Table B.2. The specific applications are mentioned respectively in the sections to follow.

The sections that follow explore the *crypto-config*, *configtx*, and *docker-compose* categories in detail. Note that, for readers not to lose track of the discussions, some information is deliberately repeated in the discussion. The following section specifically discusses the *crypto-config* category.

B.3.1.1 Crypto-config

This section explores the configuration details related to the *crypto-config.yaml* file (as shown in Figure B.4), which is a file that seeks to configure the details of all the participants. In other words, this section aims at defining all the participants of the virtual network of the proposed solution. The participants, in this case, refer to the organisations and their resources, which are nodes and users (i.e., members of the organisations). All the nodes and users belong to a specific organisation. Chapter 5 has explicitly identified all these nodes, organisations, and users in detail. Figure 5.3 identified all the users, while Figure 5.5 identified all the nodes and the organisations that might have the available resources that may be utilised. As indicated in Figure B.4, HLF uses a binary file called *cryptogen* to generate the desired network topology with all the necessary crypto materials associated with each participant and these crypto materials are stored within the *crypto-config* folder. As discussed in the previous chapter, the crypto materials are the certificates and keys used by the Blockchain network to identify all the participants and their resources.

Figure B.4 depicts the *crypto-config* category as the focus area for this section. However, to explore the configuration details related to this section, firstly, the ShareTendPro network design is discussed as an *overview of the ShareTendPro network*, which is aimed at visualising the virtual network of the proposed solution. Secondly, the configuration details contained within the *crypto-config.yaml* file is discussed. Thereafter, a *crypto-config* folder containing the cryptographic materials generated after executing the *crypto-config.yaml* file is explored later.

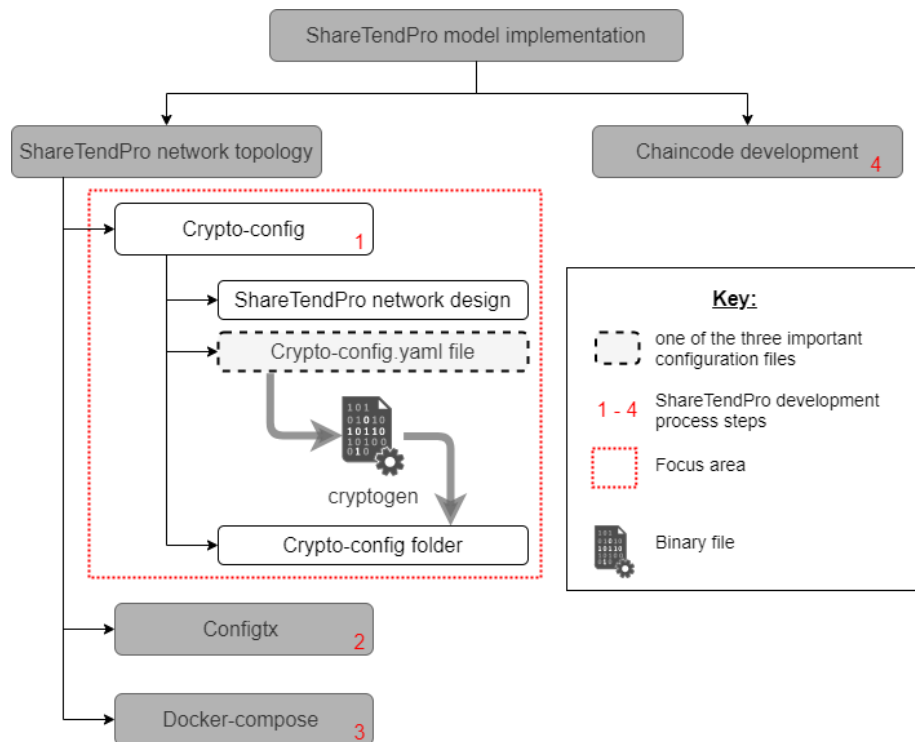


Figure B.4 Crypto-config as a focus area

B.3.1.1.1 Overview of the ShareTendPro network

The ShareTendPro network is generated in the form of a virtual network. Figure B.5 depicts the ShareTendPro network adapted from Figure B.5 in the previous chapter, however, this section only focuses on depicting the ShareTendPro network topology represented by a dark dotted-line in Figure B.5. Figure B.5 also explores how the HLF groups various nodes within the network and the grouping of these nodes are based on assigning certain nodes to a specific organisation. For instance, all the organisations in Figure B.5 (i.e., local municipality, district municipality, etc.) consist of n number of peer nodes, while the ordering service consists of four ordering nodes. Note that, to set up the virtual networking, a virtual network is created for each of the organisations. Similarly, the ordering service is also treated as an 'organisation' to set it up, like a virtual network. All these organisational networks, connected, form the entire virtual network of the proposed solution, and hence, form the Blockchain network.

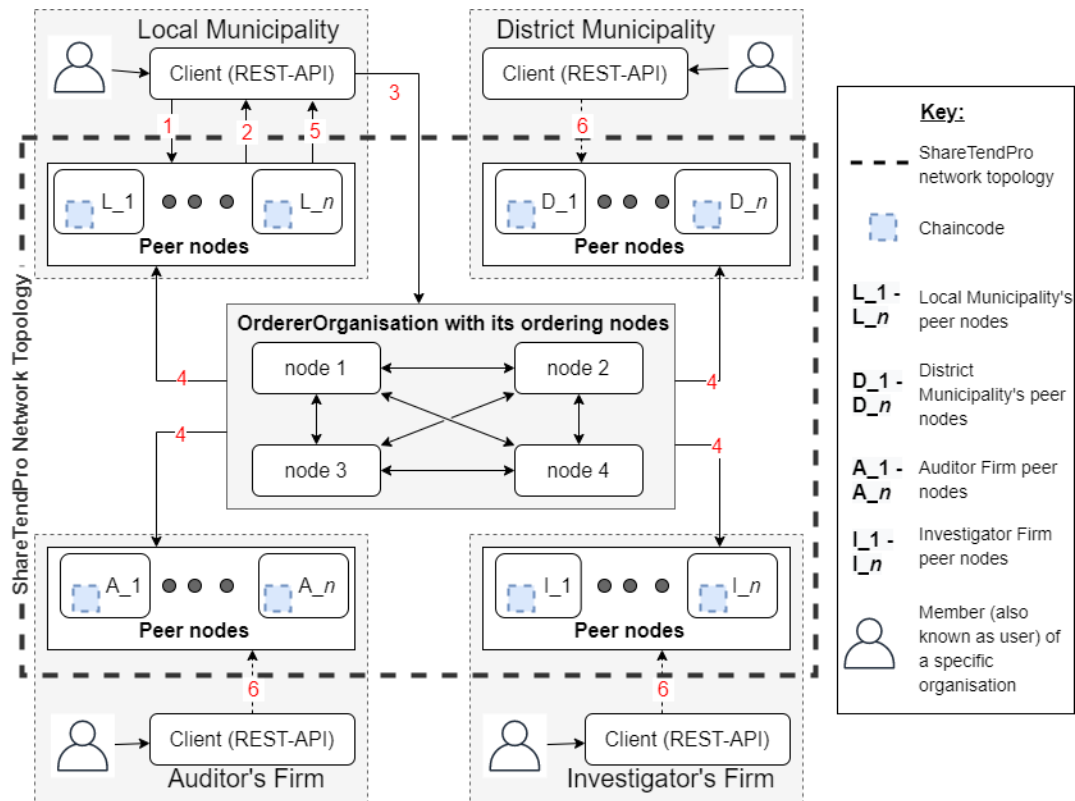


Figure B.5 ShareTendPro network topology

This section introduced the reader to the theoretical representation of the ShareTendPro network. Therefore, the following section focuses on the implementation of this network, using the *crypto-config.yaml* file.

B.3.1.1.2 Crypto-config.yaml file

This section defines all the participants (also known as organisations, the preferred term used further down) and their resources (also known as nodes and users) of the proposed solution. However, in HLF, the organisations are grouped into two types of organisations (also called definitions, the preferred term used further down) namely *OrdererOrgs* and *PeerOrgs* as shown in Figure B.6. The *OrdererOrgs* definition (represented by line 1 in Figure B.6) depicts the configuration details of the organisation of the ordering nodes (represented by node 1 to node 4 in Figure B.5), while the *PeerOrgs* definition (represented by line 19 in Figure B.6) depicts the configuration details of the organisation of the peer nodes (represented by node 1 to node n of each respective organisation in Figure B.5). Thus, the *OrdererOrgs* definition consists of one organisation called *OrdererOrganisation* (represented by line 2 in Figure B.6), while the *PeerOrgs* consist of four organisations namely *LocalMunicipality* (represented by line 19),

DistrictMunicipality (represented by line 28), *InvestigatorFirm* (represented by line 37), and *AuditorFirm* (represented by line 46).

```

1  OrdererOrgs:
2  - Name: OrdererOrganisation
3    Domain: ordererOrg.workplace
4    EnableNodeOUs: true
5    Specs:
6      - Hostname: orderer1
7        SANS:
8          - "localhost"
9      - Hostname: orderer2
10       SANS:
11         - "localhost"
12      - Hostname: orderer3
13       SANS:
14         - "localhost"
15      - Hostname: orderer4
16       SANS:
17         - "localhost"
18  PeerOrgs:
19  - Name: LocalMunicipality
20    Domain: localMun.workplace
21    EnableNodeOUs: true
22    Template:
23      Count: 2
24      SANS:
25        - "localhost"
26    Users:
27      Count: 1
28  - Name: DistrictMunicipality
29    Domain: districtMun.workplace
30    EnableNodeOUs: true
31    Template:
32      Count: 2
33      SANS:
34        - "localhost"
35    Users:
36      Count: 1
37  - Name: InvestigatorFirm
38    Domain: investigatorFir.workplace
39    EnableNodeOUs: true
40    Template:
41      Count: 2
42      SANS:
43        - "localhost"
44    Users:
45      Count: 1
46  - Name: AuditorFirm
47    Domain: auditorFir.workplace
48    EnableNodeOUs: true
49    Template:
50      Count: 2
51      SANS:
52        - "localhost"
53    Users:
54      Count: 1

```

Figure B.6 Crypto-config.yaml file structure

The mechanism of grouping these nodes into *OrdererOrgs* and *PeerOrgs* respectively enables the nodes to communicate with each other regardless of their geographical locations. This results in a distributed solution or network as discussed in Chapter 3. Additionally, the grouping of these nodes into *OrdererOrgs* and *PeerOrgs* also allows the nodes to communicate with each other using a peer-to-peer communication mechanism. The use of a peer-to-peer communication mechanism implies that the nodes of a specific organisation (e.g.,

LocalMunicipality, *DistrictMunicipality*, *InvestigatorFirm*, *AuditorFirm*, and *OrdererOrganisation*) have equal responsibility for initiating, maintaining, and terminating a specific session in a network [127].

Figure B.6 depicts the configuration details of only one *OrdererOrgs* and four *PeerOrgs*. However, to explain Figure B.6 in detail, firstly, the different elements contained within a specific organisation in Figure B.6 are explored. This is done for the elements contained within the *OrdererOrgs* definition (i.e., Name, Domain, EnableNodeOUs, and Specs), followed by the elements contained within the *PeerOrgs* definition (i.e., Name, Domain, EnableNodeOUs, Template, and Users). Thereafter, the details of all the organisations contained within Figure B.5 are explored (i.e., *OrdererOrganisation*, *LocalMunicipality*, *DistrictMunicipality*, *InvestigatorFirm*, and *AuditorFirm*).

The *OrdererOrgs* elements represented by lines 2 to 16 of Figure B.6 are as follows:

- **Name** – represents the name of the organisation that seeks to manage the ordering nodes within the proposed solution, e.g., line 2 in Figure B.6 depicts *OrdererOrganisation* as the name of the ordering service used by the proposed solution.
- **Domain** – represents the name used by the virtual network to uniquely identify the ordering service (which is the *OrdererOrganisation* in this case) and its resources – i.e., line 3 depicts *ordererOrg.workplace* as the domain name of the *OrdererOrganisation*.
- **EnableNodeOUs** – allows the virtual network to classify the roles played by various organisational units (OUs) or nodes within the network in a finer-grained manner compared to identifying them as the “members” of the Blockchain network. In other words, this element seeks to restrict the nodes of a specific organisation from accessing certain resources within the network, if it is set to true, like in the current case. However, if it were set to false, then the network would consider all the nodes or OUs of that organisation (either *OrdererOrgs* or *PeerOrgs*) as “members” of the Blockchain network. Hence, this study uses this element to identify all the ordering nodes that belong to the *OrdererOrganisation* i.e., *node 1* to *node 4* in Figure B.5.
- **Specs** – represents the default structure used by the virtual network to represent all the ordering nodes that belongs to the *OrdererOrganisation*, while the *Hostname* element within it represents the name assigned to a specific ordering node – i.e., the *hostname* of the first ordering node is *orderer1*, as seen in line 6. The subject alternative names (SANs) can be viewed as alternative domain names that might be used to access the virtual network services – i.e., the virtual network of the proposed solution runs locally,

hence, the use of *localhost* is used as a loopback mechanism that enables us to access the virtual network services. In this instance, only one SAN, i.e., *localhost*, is used. For instance, the SAN of the *orderer1* node is represented by line 8. However, Figure B.6 portrays the SANs of all the ordering nodes – i.e., the SANs of *orderer2-4* are represented by lines 11, 14, and 17 respectively.

The PeerOrgs elements represented by lines 18 to 54 of Figure B.6 are as follows:

- **Name** – represents the name of the organisation that manages the peer nodes, e.g., line 19 in Figure B.6 depicts *LocalMunicipality* as one of the organisations that manage the peer nodes within the proposed solution.
- **Domain** – represents a name used by the virtual network to uniquely identify a specific organisation that manages the peer nodes and its resources (members and nodes) – i.e., line 20 depicts *localMun.workplace* as the domain name of the *LocalMunicipality*.
- **EnableNodeOUs** – this element is the same as the EnableNodeOUs element discussed within the *ordererOrgs* elements. However, in this case, it seeks to identify all the peer nodes that belong to an organisation that manages the peer nodes, i.e., nodes L_1 to L_n in Figure B.5.
- **Template** – represents the default structure used to group all the peer nodes of an organisation that seeks to manage the peer nodes, while the *Template.Count* element within it represents the total number of peer nodes (2 in this case, i.e., L_1 and L_2) created for that organisation.
- **Users** – represent the default structure used to group the users that belong to a specific organisation that seek to be part of the proposed solution, while the *User.Count* element within it represents the number of users created for that organisation, excluding the default user (which is the *Admin* user). However, a ‘default user’, i.e., the *Admin* user, is added by default by the HLF framework hence, all the organisations within PeerOrgs are comprised of two users in total, i.e. *user1* and the *Admin* user respectively. The total number of users corresponds with the total number of nodes created for a specific organisation within the virtual network and they are using a *client (REST-API)* node to interact with the virtual network, as shown in Figure B.5. This *client* node might be viewed as a normal computer that runs an application that is capable of interacting with the virtual network of the proposed solution using REST-API, as indicated in Chapter 5.

The details of all the organisations contained within Figure B.5 are discussed next:

- **OrdererOrganisation** (which is the only name of the *OrdererOrgs* definition, as seen in line 2 of Figure B.6) consists of three ordering nodes namely *orderer1* (represented by line 6 within the *hostname* element), *orderer2* (represented by line 9), *orderer3* (represented by line 12), and *orderer4* (represented by line 15). The domain name of OrdererOrganisation is *ordererOrg.workplace* (as seen in line 3).
- **LocalMunicipality** (which is the name of the first organisation of the *PeerOrgs* definition, as seen in line 19 of Figure B.6) consists of two peer nodes represented by the *Template.Count* element (as shown in line 23). The SAN for these peer nodes is *localhost* (as seen in line 25), i.e., the two peer nodes from Figure B.5 are referred to as *L_1* and *L_2* in this case. The domain name of this organisation is *localMun.workplace* (as shown in line 20). Additionally, the LocalMunicipality organisation consists of one user represented by the *Users.Count* element (as seen in line 27).
- **DistrictMunicipality** (which is the name of the second organisation of the *PeerOrgs* definition, as seen in line 28 of Figure B.6) also consists of two peer nodes and one user, like the LocalMunicipality organisation. However, the *Template* elements (represented by lines 31 to 34), as well as the *Users* element (represented by line 36) for DistrictMunicipality, are the same as the ones discussed within the LocalMunicipality. The domain name for DistrictMunicipality is *districtMun.workplace*, as shown in line 29.
- **InvestigatorFirm** (which is the name of the third organisation of the *PeerOrgs* definition, as seen in line 37 of Figure B.6) also consists of two peer nodes and one user, like the other organisations. The *Template* elements (represented by lines 40 to 43), as well as the *Users* element (represented by line 45) for InvestigatorFirm, are the same as the ones discussed within the LocalMunicipality. The domain name for InvestigatorFirm is *investigatorFir.workplace*, as shown in line 38.
- **AuditorFirm** (which is the name of the fourth organisation of the *PeerOrgs* definition, as seen in line 46 of Figure B.6) also consists of two peer nodes and one user, like the other organisations. Hence, the *Template* elements (represented by lines 49 to 52), as well as the *Users* element (represented by line 54) for AuditorFirm, are the same as the ones discussed within the LocalMunicipality. The domain name for AuditorFirm is *auditorFir.workplace*, as shown in line 47.

This section explored all the necessary information required to configure the desired ShareTendPro network topology. Therefore, the following section explores the results generated after executing the *crypto-config.yaml* file using a binary file called *cryptogen*, and these results are stored within the *crypto-config* folder as shown in Figure B.4.

B.3.1.1.3 Crypto-config folder

This section examines the results generated after executing the *crypto-config.yaml* file. These results are nothing, but the cryptographic materials (also known as cryptographic libraries) used by the virtual network to either encrypt, decrypt or authenticate various organisations (*LocalMunicipality*, *DistrictMunicipality*, *InvestigatorFirm*, *AuditorFirm*, and *OrdererOrganisation*) and their resources (e.g. nodes, or users) as they interact with the Blockchain network. Hence, this section seeks to explore how the Blockchain network groups these crypto materials within the *crypto-config* folder.

Figure B.7 depicts all the crypto materials required to secure the communication within the virtual network of the proposed solution. However, these crypto materials are grouped into two categories namely: *ordererOrganizations* and *peerOrganizations* as shown in Figure B.7. The *ordererOrganizations* category depicts all the crypto materials of the organisation that belong to the *OrdererOrgs* definition, which is called *OrdererOrganisation* as indicated in the previous section. The *peerOrganizations* category depicts all the materials of the organisations that belong to the *PeerOrgs* definition (*LocalMunicipality*, *DistrictMunicipality*, *InvestigatorFirm*, and *AuditorFirm*). However, all the crypto materials are identified using a domain name of a specific organisation that seeks to be part of the proposed solution. For instance, the crypto materials of the only one *ordererOrganizations* are stored within the *ordererOrg.workplace* folder (represented by label 2 of Figure B.7), whereby the *ordererOrg.workplace* depicts the domain name of the organisation called *OrdererOrganisation* as shown in Figure B.6. The crypto materials of one of the organisations that belong to the *peerOrganizations* category are stored within the *localMun.workplace* folder (represented by label 5 in Figure B.7), whereby the *localMun.workplace* represents the domain name of the organisation called *LocalMunicipality* in Figure B.6. The same notion might also be applied to the following domain names *districtMun.workplace*, *investigatorFir.workplace*, and *auditorFir.workplace*. The folder structuring within the two categories *ordererOrganizations* and *peerOrganizations* are similar, except when it comes to grouping the nodes within a specific folder. For example, the organisation within the

ordererOrganizations category groups its nodes within the orderers' folder (i.e., label 2.c in Figure B.7), while the organisation within the *peerOrganizations* category groups the nodes within the peers' folder (i.e., label 5.c in Figure B.7). Figure B.7 is now explained in detail.

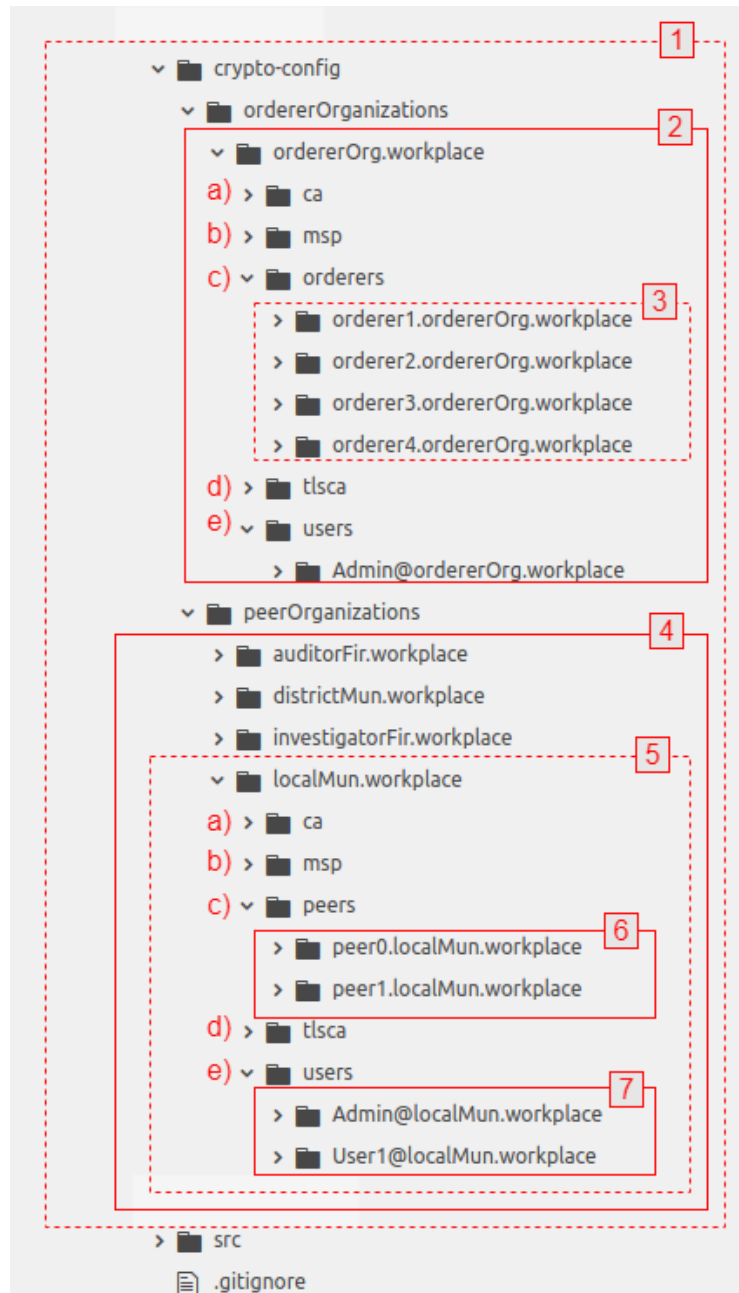


Figure B.7 Crypto-config folder

The following items examine the folder structure used to group all the crypto materials in Figure B.7:

- **Label 1** – explores the boundaries of the crypto-config folder. In other words, this item seeks to represent all the crypto materials or materials that belong to the crypto-config folder.

- **Label 2** – represents all the crypto materials that belong to the *ordererOrganizations* category which consists of the folder structuring of the only one ordering service with a domain name *ordererOrg.workplace* as follows:
 - a. *CA (certificate authority)* – this folder contains the crypto materials that allow the virtual network to issue out the identities of all the resources (i.e., nodes or users) that belong to the *ordererOrg.workplace* organisation.
 - b. *MSP (membership service provider)* – this folder contains the crypto materials used by the virtual network to identify all the resources (i.e., nodes or users) that belong to the *ordererOrg.workplace* organisation.
 - c. *Orderers* – this folder contains all the crypto materials of all the ordering nodes that belong to a domain name *ordererOrg.workplace* (also represented by label 3 in Figure B.7). Additionally, each folder within the *orderers*’ folder contains the crypto materials of a specific ordering node, e.g. the crypto materials of the first ordering node are contained within the *orderer1.ordererOrg.workplace* folder. This notion might also be applied to the remaining folders *orderer2.ordererOrg.workplace*, *orderer3.ordererOrg.workplace*, and *orderer4.ordererOrg.workplace*.
 - d. *TLSCA (transport layer security certificate authority)* – this folder contains the crypto materials used by an organisation within a domain name *ordererOrg.workplace* and its resources (i.e., ordering nodes or users) to secure the communication channel as they interact within the Blockchain network.
 - e. *Users* – this folder contains the crypto materials of all the users that have the administrative rights to perform certain tasks using the *ordererOrg.workplace* organisation. However, in this case, only the *Admin* user (represented by *Admin@ordererOrg.workplace*) has the right to perform these administrative tasks within the network.
- **Label 3** – as indicated earlier on, this label identifies the crypto materials of the four ordering nodes that belong to a domain name *ordererOrg.workplace*, and these ordering nodes are *orderer1.ordererOrg.workplace*, *orderer2.ordererOrg.workplace*, *orderer3.ordererOrg.workplace*, and *orderer4.ordererOrg.workplace*.
- **Label 4** – represents the crypto materials of all the organisations (LocalMunicipality, DistrictMunicipality, InvestigatorFirm, and AuditorFirm) that seek to form part of the proposed solution. However, the crypto materials of these organisations are grouped

using their domain name as shown in Figure B.7. For instance, the crypto materials of one of the organisations that belong to the *peerOrganizations* category are stored within the domain name *localMun.workplace*, which is the LocalMunicipality. This notion might also be applied to the following domain names: *districtMun.workplace*, *investigatorFir.workplace*, and *auditorFir.workplace*, but these are not shown in Figure B.7 to save space.

- **Label 5** – depicts the boundaries of the crypto materials of an organisation within a domain name *localMun.workplace*, which is the LocalMunicipality. The grouping of all the crypto materials that belong to this organisation is almost the same as the grouping of the crypto materials of an organisation with a domain name *ordererOrg.workplace*, as discussed in label 2. However, the main difference between the two is the grouping of nodes since the *localMun.workplace* organisation groups its nodes within the *peers*' folder instead of grouping them into an *orderers*' folder. Therefore, the following items seek to examine the structuring of these crypto materials within the *localMun.workplace* organisation:
 - a. *CA* – it is the same as the item discussed in label 2.a, except that this time around these crypto materials allow the virtual network to issue the identities of the resources that belong to the *localMun.workplace* organisation.
 - b. *MSP* – it is the same as label 2.b, except that this time around it depicts the membership of all the resources (peer nodes or users) that belong to the *localMun.workplace* organisation.
 - c. *Peers* – this folder contains the crypto materials of all the peer nodes that belong to the *localMun.workplace* organisation (as shown in label 6). Additionally, each folder within the *peers*' folder contains the crypto materials of a specific peer node, e.g. the crypto materials of the first peer node are contained within the *peer0.localMun.workplace* folder, while the crypto materials of the second peer node are contained within the *peer1.localMun.workplace* folder.
 - d. *TLSCA* – it is the same as label 2.d, except that this time around it seeks to secure the communication channel when the *localMun.workplace* organisation and its resources (nodes and users) interact with the Blockchain network.
 - e. *Users* – it is the same as label 2.e, except that it depicts the crypto materials of all the users that belong to the *localMun.workplace* organisation.

- **Label 6** – as indicated earlier on, this label represents the crypto materials of all the peer nodes that belong to the *local.workplace* organisation. There are two peer nodes that belong to the *localMun.workplace* organisation namely *peer0* and *peer1* – hence, the crypto materials of these two peer nodes are represented as *peer0.localMun.workplace* and *peer1.localMun.workplace*.
- **Label 7** – depicts the crypto materials of all the users that belong to the *localMun.workplace* organisation and there are two users within this organisation namely *Admin* (represented by *Admin@localMun.workplace*) and *User1* (represented by *User1@localMun.workplace*).

This section detailed all the necessary information required to generate the crypto materials of the desired network topology (i.e. the ShareTendPro network). Therefore, the following section focuses on configuring the communication channel that would be used by various organisations to secretly share project information within the Blockchain network.

B.3.1.2 Configtx

This section explores the configuration details related to the *configtx.yaml* file as shown in Figure B.8.

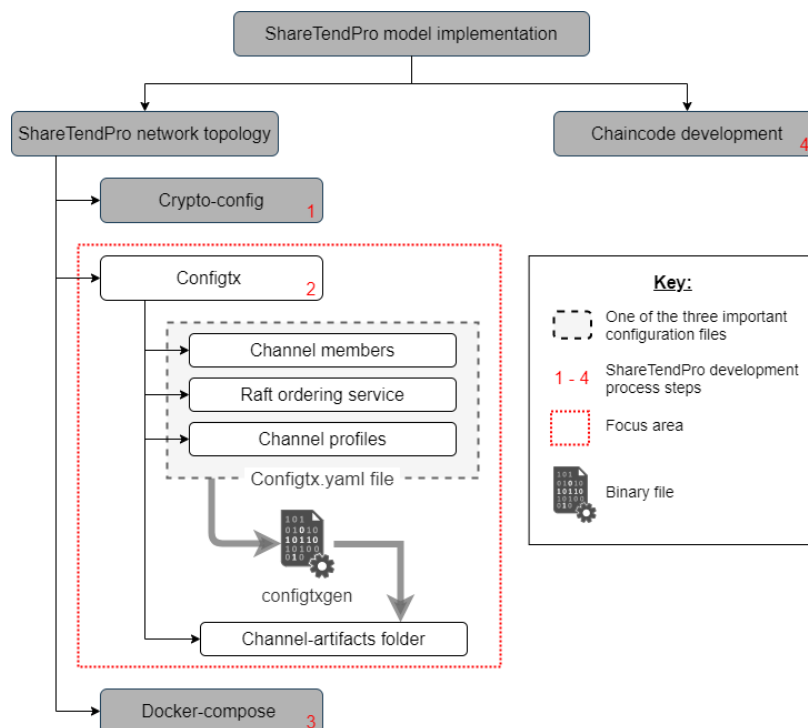


Figure B.8 Configtx as a focus area

This section seeks to explore the configuration details that aimed at generating the communication channel that will be used by various organisations to secretly share project information. As indicated in Figure B.8 this section makes use of a tool called YAML library and a library file called *configtxgen*. The configuration details contained within the *configtx.yaml* file are compiled using the YAML library, which is then fed to a *configtxgen* binary file to generate the results (also known as channel-artifacts) which are stored within the channel-artifacts folder, as shown in Figure B.8. The channel-artifacts are nothing, but the necessary details required to generate a private communication channel, hence, the use of a private communication channel results in a private Blockchain network as discussed in Chapter 3.

Figure B.8 depicts the *configtx* category as the focus area for this section. Hence, this section divided the configuration details contained within the *configtx.yaml* file into three subsections namely: *channel members*, *Raft ordering service*, and *channel profiles*. The first subsection explores the configuration details related to the members of the communication channel, which are various organisations that seek to be part of the proposed solution as *channel members*. The second subsection explores the configuration details related to the chosen type of ordering service used to add new transactions to the virtual network of the proposed solution, as the *Raft ordering service*. The last subsection explores the configuration details related to the profile used to generate the results or channel artifacts that would be used by the virtual network of the proposed solution, as *channel profiles*. Thereafter, the results contained within the channel-artifacts folder are discussed later.

B.3.1.2.1 Channel members

This section explores the configuration details of all the organisations that form part of the channel members of the proposed solution. A general overview of the channel members is first provided, followed by a more detailed discussion. Hence, the channel members consist of five organisations namely: *OrdeerOrganisationMSP*, *LocalMunicipalityMSP*, *DistrictMunicipalityMSP*, *InvestigatorFirmMSP*, and *AuditorFirmMSP*. Remember that ‘MSP’ within each of these channel members stands for membership service provider as shown in Figure B.7. However, there are certain elements that are used to identify all the organisations that belong to a particular communication channel and these elements are *ID* and *MSPDir*, as shown in Figure B.9. The *ID* element is used to uniquely identify an organisation (member) within the channel, while the *MSPDir* is used to locate the MSP crypto materials associated

with that organisation. For instance, line 4 of Figure B.9 depicts *OrdererOrgMSP* as the *ID* of a channel member called *OrdererOrganisationMSP*, while line 5 references the MSP crypto materials of the domain name *ordererOrg.workplace*, which is the crypto materials associated with the *OrdererOrganisation* in Figure B.9, label 2.b. A more detailed discussion of the channel members configuration as shown in Figure B.9 follows next.

```

1 Organizations:
2   - &OrdererOrg
3     Name: OrdererOrganisation
4     ID: OrdererOrgMSP
5     MSPDir: crypto-config/ordererOrganizations/ordererOrg.workplace/msp
6     Policies:
7       Readers:
8         Type: Signature
9         Rule: "OR('OrdererOrgMSP.member')"
10      Writers:
11        Type: Signature
12        Rule: "OR('OrdererOrgMSP.member')"
13      Admins:
14        Type: Signature
15        Rule: "OR('OrdererOrgMSP.admin')"
16   - &LocalMun
17     Name: LocalMunicipalityMSP
18     ID: LocalMunMSP
19     MSPDir: crypto-config/peerOrganizations/localMun.workplace/msp
20     Policies:
21       Readers:
22        Type: Signature
23        Rule: "OR('LocalMunMSP.admin', 'LocalMunMSP.peer', 'LocalMunMSP.client')"
24       Writers:
25        Type: Signature
26        Rule: "OR('LocalMunMSP.admin', 'LocalMunMSP.client')"
27       Admins:
28        Type: Signature
29        Rule: "OR('LocalMunMSP.admin')"
30       Endorsement:
31        Type: Signature
32        Rule: "OR('LocalMunMSP.peer')"
33     AnchorPeers:
34       - Host: peer0.localMun.workplace
35         Port: 7051
36   - &DistrictMun
37     Name: DistrictMunicipalityMSP
38     ID: DistrictMunMSP
39     MSPDir: crypto-config/peerOrganizations/districtMun.workplace/msp
40     Policies:
41       Admins:
42        Type: Signature
43        Rule: "OR('DistrictMunMSP.admin')"
44     AnchorPeers:
45       - Host: peer0.districtMun.workplace
46         Port: 9051
47   - &InvestigatorFir
48     Name: InvestigatorFirmMSP
49     ID: InvestigatorFirmMSP
50     MSPDir: crypto-config/peerOrganizations/investigatorFir.workplace/msp
51     Policies:
52       Admins:
53        Type: Signature
54        Rule: "OR('InvestigatorFirmMSP.admin')"
55     AnchorPeers:
56       - Host: peer0.investigatorFir.workplace
57         Port: 11051
58   - &AuditorFir
59     Name: AuditorFirmMSP
60     ID: AuditorFirmMSP
61     MSPDir: crypto-config/peerOrganizations/auditorFir.workplace/msp
62     Policies:
63       Admins:
64        Type: Signature
65        Rule: "OR('AuditorFirmMSP.admin')"
66     AnchorPeers:
67       - Host: peer0.auditorFir.workplace
68         Port: 13051

```

Figure B.9 Members of the channel

The following items explore the configuration details of all the channel members as follows:

- **OrdererOrg** (which is the name of the first member of the channel, as shown in line 2 of Figure B.9) depicts *OrdererOrganisationMSP* as the name of this channel member, as seen in line 3. The ID element of this channel member is *OrdererOrgMSP*, as shown in line 4, while the *MSPDir* element of this channel member references the crypto materials associated with a domain name *ordererOrg.workplace* (as seen in line 5), which is the crypto materials of an organisation called *OrdererOrganisationMSP*. Additionally, line 6 depicts various policies (i.e., *Readers*, *Writers*, and *Admins*) associated with this member of the channel. For instance, lines 7-9 represent a *Readers* policy that allows all the resources (ordering nodes and users) that belong to the *OrdererOrganisationMSP* to read or access the information from the Blockchain network, while line 8 represents the type of policy used by the proposed solution, which is a ‘signature’ policy. For example, all the members of the *OrdererOrganisationMSP* (which are the ordering nodes and the *admin* user) are required to digitally sign their transactions whenever they are interacting with the Blockchain network, and this mechanism also allows the network to determine all the authorised members or resources. The signature policy can be viewed as a policy that might only be satisfied by the signature of an identity role of certain resources (e.g., nodes or users) within the Blockchain network. Lines 10-12 depicts a *Writers* policy that allows all the resources that belong to this channel member to write information (add new transactions) to the Blockchain network, while lines 13-15 represents an *Admins* policy that allows only the *admin* user to perform the administrative tasks associated with this channel member.
- **LocalMun** (which is the name of the second channel member, as shown in line 16 of Figure B.9) depicts *LocalMunicipalityMSP* as the name of the channel member, as seen in line 17. The *ID* and *MSPDir* elements highlighted within this channel member are like the ones discussed within the *OrdererOrg* member, hence, these elements are not explained in detail again. Only the additional elements that were not discussed before are explained in detail. Line 20 depicts various policies (i.e., *Readers*, *Writers*, *Admins*, and *Endorsement*) associated with this channel member. Lines 21-23 depicts a *Readers* policy that allows the *admin*, *peer*, and *client* resources that belong to this channel member to read or access information from the Blockchain network. Lines 24-26 represents a *Writers* policy that allows the *admin* and *client* resources to write information to the Blockchain network. Lines 27-29 depicts an *Admins* policy that allows only the admin resource to perform the administrative tasks associated with this

channel member, while lines 30-32 depicts an *Endorsement* policy that allows peer resources to endorse transactions submitted by writers (which are the admin and client resources). Additionally, lines 33-35 depicts an element that seeks to configure the details of an anchor peer assigned to this channel member, which is *peer0.localMun.workplace* (represented by line 34) with an exposed port of 7051 (represented by line 35). All the elements discussed within this channel member are like the ones highlighted within the *DistrictMun* (represented by lines 36-55), *InvestigatorFir* (represented by lines 56-75), and *AuditorFir* (represented by lines 76-96). Hence, the configuration details of these channel members are not explored further.

This section discussed the configuration details of all the channel members that would be part of the private communication channel. Therefore, the following section explores the type of ordering services used by the proposed solution to add new transactions to the Blockchain network.

B.3.1.2.2 Raft ordering service

This section focuses on the configuration details in relation to the type of ordering service used by the proposed solution. A general overview of the Raft ordering service is first provided, followed by a more detailed discussion. Line 200 of Figure B.10 depicts the type of ordering service used by the proposed solution, which is *etcdraft*. The *etcdraft* is a combination of two words namely *etcd* and *raft*. The first word “*etcd*” can be viewed as a distributed key-value store that seeks to manage critical information in a distributed system [128] [129]. Some of this critical information include configuration data and state data (information stored within a Blockchain section in a ledger).

The second word “*raft*” can be viewed as a consensus algorithm that seeks to maintain consistency of the data stored across all the nodes within a raft cluster [128] [130]. A raft cluster can be viewed as a group of nodes connected in a distributed manner – for instance, a raft cluster of the proposed solution is formed using four ordering nodes represented by node 1 to node 4 in Figure B.5. Additionally, the consistency of the data stored within the raft cluster is achieved by electing a leading node (ordering node in this case, e.g., assume that node 3 in Figure B.5 was elected) that would be responsible for managing replicas of the data for other nodes in a cluster, which are called followers (e.g., node 1, 2, and 4 in Figure B.5). For example, if the transaction received by node 2 is compromised (altered), then the leading node (which is node 3) is going to reject it because three nodes (which are node 1, 3, and 4) would have reached

a consensus regarding a particular piece of data, while one node has reached its consensus on a different piece of data – hence, the raft cluster might also be used to maintain data integrity across the network since the data would be approved by three nodes and rejected by one node. Therefore, the responsibility of the leading node (e.g., node 3 in this case) is to accept the transactional requests submitted by the Client (REST-API) and forward them to follower nodes. However, if one of the follower nodes (e.g., node 1) fails to receive a message from the leading node (which is node 3) at a given time frame due to network connectivity reasons, then a voting mechanism is initiated by node 1 to elect a new leader. This voting mechanism starts when a follower node (node 1 in this case) declares itself as a candidate, which allows other nodes to vote for it. Once that candidate receives two-thirds majority votes then the Raft cluster will declare it as the new leader that would be responsible for managing replicas of the data for other nodes in a cluster – the process only works whenever the Blockchain network of the proposed solution is up and running¹ [130]. A more detailed discussion of the Raft ordering service configuration as shown in Figure B.10 follows next.

```

198 Orderer: &OrdererDefaults
199 # Orderer Type: The orderer implementation to start
200 OrdererType: etcdraft
201 EtcdRaft:
202   Consenters:
203     - Host: orderer1.ordererOrg.workplace
204       Port: 7050
205       ClientTLS Cert: crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer1.ordererOrg.workplace/tls/server.crt
206       ServerTLS Cert: crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer1.ordererOrg.workplace/tls/server.crt
207     - Host: orderer2.ordererOrg.workplace
208       Port: 8050
209       ClientTLS Cert: crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer2.ordererOrg.workplace/tls/server.crt
210       ServerTLS Cert: crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer2.ordererOrg.workplace/tls/server.crt
211     - Host: orderer3.ordererOrg.workplace
212       Port: 9050
213       ClientTLS Cert: crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer3.ordererOrg.workplace/tls/server.crt
214       ServerTLS Cert: crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer3.ordererOrg.workplace/tls/server.crt
215     - Host: orderer4.ordererOrg.workplace
216       Port: 10050
217       ClientTLS Cert: crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer4.ordererOrg.workplace/tls/server.crt
218       ServerTLS Cert: crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer4.ordererOrg.workplace/tls/server.crt

```

Figure B.10 Raft ordering service

Figure B.10 depicts the configuration details of a Raft ordering service that consist of four ordering nodes as follows:

- **Orderer1.ordererOrg.workplace** (represented by line 203) depicts the name of the first ordering node of the Raft cluster. The exposed port used by this ordering node is 7050 (as shown in line 204). Additionally, lines 205 and 206 references the crypto materials used by this ordering node to secure the communication channel as it interacts with the Blockchain network. The crypto materials used by line 205 and 206 is a *server.crt*, which represent a server certificate used in the crypto procedure.

¹ A fully detailed explanation and simulation of how the Raft ordering service work, is found at <https://raft.github.io/>, for a clear visual animation of how the entire process work.

- **Orderer2.ordererOrg.workplace** (represented by lines 207 to 210) depicts the configuration details which are similar to the ones discussed within the *Orderer1.ordererOrg.workplace*. This notion also applies to the configuration details of the following ordering node services namely: *Orderer3.ordererOrg.workplace*, and *Orderer4.ordererOrg.workplace*. Hence, these details are not explained further to avoid the repetition of some of the concepts.

This section has explored all the configuration details related to the Raft ordering service. Therefore, the following section discusses the configuration details related to the channel profiles used to generate the results which are stored in the channel-artifacts folder (which is discussed later).

B.3.1.2.3 Channel profiles

As indicated in Figure B.8, the *configtx.yaml* file is fed to a binary file called *configtxgen* and the results (also known as channel artifacts) are stored within the channel-artifacts folder. Technically, the *configtxgen* binary file makes use of the channel profiles to generate all the necessary channel artifacts required to create a private communication channel for the proposed solution. Hence, Figure B.11 depicts the configuration details of the two-channel profiles namely: *BasicChannel* profile (represented by line 297) and *OrdererGenesis* profile (represented by line 310). A more detailed discussion of the channel profiles configuration as shown in Figure B.11 follows next.

```

295 Profiles:
296
297 BasicChannel:
298 Consortium: SampleConsortium
299 <<: *ChannelDefaults
300 Application:
301 <<: *ApplicationDefaults
302 Organizations:
303 - *LocalMun
304 - *DistrictMun
305 - *InvestigatorFir
306 - *AuditorFir
307 Capabilities:
308 <<: *ApplicationCapabilities
309
310 OrdererGenesis:
311 <<: *ChannelDefaults
312 Capabilities:
313 <<: *ChannelCapabilities
314 Orderer:
315 <<: *OrdererDefaults
316 Organizations:
317 - *OrdererOrg
318 Capabilities:
319 <<: *OrdererCapabilities
320 Consortiums:
321 SampleConsortium:
322 Organizations:
323 - *LocalMun
324 - *DistrictMun
325 - *InvestigatorFir
326 - *AuditorFir

```

Figure B.11 Channel profiles

The following items explore the configuration details contained within the two profiles as follows:

- **BasicChannel** (which is the name of the first profile as shown in line 297 of Figure B.11) depicts the configuration details of a communication channel that consists of four organisations namely: *LocalMun* (represented by line 303), *DistrictMun* (represented by line 304), *InvestigatorFir* (represented by line 305), and *AuditorFir* (represented by line 306). However, the “ * “ symbol before each of these organisations references the configuration details discussed within the channel members section or Figure B.9. The other information contained within this channel profile forms part of the default configuration details, hence, it is not important to be discussed for this research.
- **OrdererGenesis** (which is the name of the second profile as shown in line 310 of Figure B.11): depicts the configuration details of a profile that seeks to generate the genesis block of the Blockchain network. As indicated in Chapter 3, the genesis block is the first block within the Blockchain network, hence, this profile seeks to generate the transactions that constitute the genesis block. Additionally, this profile allows the Blockchain network to distribute the genesis block to all the nodes (peer nodes and ordering nodes) that form part of the proposed solution. Line 321 depicts the

configuration details of the consortium that allows this profile to share the genesis block with all the organisations (participants) that seek to be part of the proposed solution, while line 317 depicts the configuration details that allows this profile to share the genesis block with the organisation that handles the ordering service, which is the *OrdererOrganisation* as discussed in Figure B.9. The other information or configuration details highlighted within this profile form part of the default configurations, hence, it is not important to be discussed for this research.

This section has explored all the necessary information required to create a private communication channel that might be used by various participants (Local Municipality, District Municipality, Investigator’s Firm, and Auditor’s Firm) to secretly share project information among each other. Therefore, the following section examines the results generated after executing the *configtx.yaml* file using a binary file called *configtxgen*, and these results are stored within the *channel-artifacts* folder as shown in Figure B.8.

B.3.1.2.4 Channel-artifacts folder

This section examines the results generated after executing the *configtx.yaml* file. These results can also be viewed as the channel artifacts since they are part of the materials used by the Blockchain network to generate a communication channel that would be used by various participants to share project information.

Figure B.12 depicts all the channel artifacts generated after executing the *configtx.yaml* file. *Label C* in Figure B.12 depict a channel artifact generated using a profile called *OrdererGenesis*, while the other channel artifacts (represented by *Label A, B, D, E, and F*) were generated using a profile called *BasicChannel*. *Label F* seems to be the odd-one-out here, however, more explanation follows below.

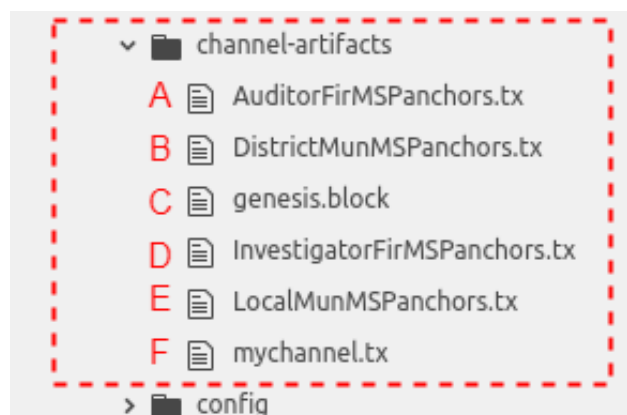


Figure B.12 Channel-artifacts folder

The following items seek to explore the details of these channel artifacts:

- **Label A** represents a channel artifact that contains all the necessary information required by the Blockchain network to identify an anchor peer node that belongs to the Auditor's Firm. Hence, the file name for this channel artifact is *AuditorFirMSPanchors.tx*.
- **Label B, D, and E** contains similar information as discussed within Label A, hence, it is not explained again.
- **Label C** represents a channel artifact that contains all the necessary information required by the Blockchain network to identify the genesis block. Hence, the file name for this channel artifact is *genesis.block*. Additionally, this channel-artifact contains all the necessary information required to identify all the nodes (peer nodes and ordering nodes) that form part of the Blockchain network, as discussed in the previous chapter.
- **Label F** depicts a channel artifact that contains all the necessary information required by the Blockchain network to identify the communication channel used by the proposed solution. Hence, the file name for this channel artifact is *mychannel.tx*. This channel-artifact is different from the previous channel-artifacts because it contains all the necessary information that seeks to identify the organisations (e.g., Local Municipality, District Municipality, Investigator's Firm, and Auditor's Firm) that might be interested in being part of the Blockchain network. Additionally, it also contains information that seeks to identify the corresponding file (e.g., Labels A, B, D, or E) that contains the configuration of the anchor peer of each of the respective organisations.

This section has discussed details required to generate a private communication channel with all the necessary channel artifacts that might be used to secretly share project information. Therefore, the following section explores the configuration details related to the third important file, which is the *docker-compose.yaml* file. This file contains the configuration details that seek to generate the virtual nodes (also known as the network services) that would be used by the proposed solution.

B.3.1.3 Docker-compose

This section explores the configuration details related to the third file, which is *docker-compose.yaml*, as shown in Figure B.13. In other words, this section explores the configuration details of all the services used by the Blockchain network. A service in this case might be viewed as a docker container because it packages all the necessary information required to

perform a specific task into one package that can be deployed to the Blockchain network as a single package or program. For instance, a specific node (i.e., ordering node 1) can be viewed as a single package that renders certain services to the Blockchain network since it performs part of the functionalities that seek to add a new block of transactions to the network.

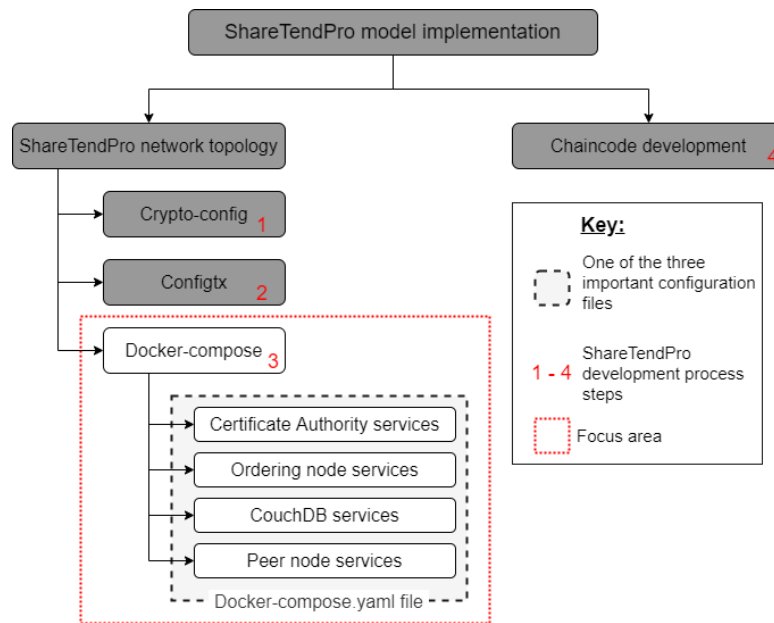


Figure B.13 Docker-compose as the focus area

Figure B.13 depicts all the services required by the proposed solution. A general overview of these services is firstly provided, followed by a more detailed discussion. This section classified these services into seven different categories namely: *Certificate Authority services*, *Ordering node services*, *CouchDB services*, and *Peer node services*. Therefore, the following sections explore the configuration details contained within each of these services, starting with the configuration details of the *Certificate Authority services* because it aimed at issuing the digital certificates of all the resources (i.e., nodes or users) that belongs to a specific organisation. The second section to follow focuses on the configuration details of the *Ordering node services*, which portrays the services used by the proposed solution to add a new block of transactions to the ShareTendPro network. Thereafter, the third section discusses the configuration details of the services that seek to store data within the Blockchain network, which is the *CouchDB services*. The last section to follow depicts the configuration details of the Peer node services.

B.3.1.3.1 Certificate Authority services

This section explores the configuration details of all the certificate authority services as shown in Figure B.14, which are *ca-localMun* (represented by line 7), *ca-districtMun* (represented by

line 28), *ca-investigatorFir* (represented by line 49), and *ca-auditorFir* (represented by line 70). The CA within each of these services stands for *certificate authority*, which is like the CA discussed within Figure B.7. A more detailed discussion of the certificate authority services configuration as shown in Figure B.14 follows.

```

6  services:
7  ca-localMun:
8      image: hyperledger/fabric-ca
9      environment:
10         - FABRIC_CA_HOME=/etc/hyperledger/fabric-ca-server
11         - FABRIC_CA_SERVER_CA_NAME=ca.localMun.workplace
12         - FABRIC_CA_SERVER_CA_CERTFILE=/etc/hyperledger/fabric-ca-server-config/ca.localMun.workplace-cert.pem
13         - FABRIC_CA_SERVER_CA_KEYFILE=/etc/hyperledger/fabric-ca-server-config/priv_sk
14         - FABRIC_CA_SERVER_TLS_ENABLED=true
15         - FABRIC_CA_SERVER_TLS_CERTFILE=/etc/hyperledger/fabric-ca-server-tls/tlsca.localMun.workplace-cert.pem
16         - FABRIC_CA_SERVER_TLS_KEYFILE=/etc/hyperledger/fabric-ca-server-tls/priv_sk
17     ports:
18         - "7054:7054"
19     command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
20     volumes:
21         - ./channel/crypto-config/peerOrganizations/localMun.workplace/ca:/etc/hyperledger/fabric-ca-server-config
22         - ./channel/crypto-config/peerOrganizations/localMun.workplace/tlsca:/etc/hyperledger/fabric-ca-server-tls
23     container_name: ca.localMun.workplace
24     hostname: ca.localMun.workplace
25     networks:
26         - test
27
28     ca-districtMun:
29         image: hyperledger/fabric-ca
30         environment:
31             ports:
32                 - "8054:7054"
33             command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
34             volumes:
35                 container_name: ca.districtMun.workplace
36                 hostname: ca.districtMun.workplace
37             networks:
38                 - test
39
40     ca-investigatorFir:
41         image: hyperledger/fabric-ca
42         environment:
43             ports:
44                 - "9054:7054"
45             command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
46             volumes:
47                 container_name: ca.investigatorFir.workplace
48                 hostname: ca.investigatorFir.workplace
49             networks:
50                 - test
51
52     ca-auditorFir:
53         image: hyperledger/fabric-ca
54         environment:
55             ports:
56                 - "10054:7054"
57             command: sh -c 'fabric-ca-server start -b admin:adminpw -d'
58             volumes:
59                 container_name: ca.auditorFir.workplace
60                 hostname: ca.auditorFir.workplace
61             networks:
62                 - test

```

Figure B.14 Certificate Authority services

The following items explore the configuration details of all the certificate authority services highlighted within Figure B.14 as follows:

- **CA-localMun:** depicts the name of the first certificate authority service, represented by lines 7-26. Line 8 depicts an image that contains all the dependencies required by this service, which is *fabric-ca*. Lines 9-16 represent all the environmental variables required by this service – whereby lines 11-13 depict the variables used by the server, while lines 14-16 depict the variables used to secure the communication channel. Lines

17 and 18 represent the exposed port used by this service, which is 7054. Line 19 depicts an element that contains the command that seeks to run this service. Lines 20-22 represent the crypto materials used by this service, which references the crypto materials of a domain name *localMun.workplace*, which is the *Local Municipality*. Lines 23 and 24 represent the docker container name and the hostname as *ca.localMun.workplace*. The naming of these variables might be different; however, this study uses the same name as part of trying to simplify the logic behind identifying the docker container within the development side and the hostname within the virtual network. Using the same name also simplifies some of the configuration details especially when it comes to debugging the ShareTendPro model since some of the configurations requires only the docker container name, while others require only the hostname. Hence, the use of the same name eliminates issues related to figuring out which variable is required for a specific configuration. The docker container name is used to identify this container during the execution of all the services, while the hostname is used to identify this service within the Blockchain network. Line 25 and 26 represent the name assigned to the virtual network of the proposed solution, which is the *test*.

- **CA-districtMun** (represented by lines 28-47), **CA-investigatorFir** (represented by lines 49-68), and **CA-auditorFir** (represented by lines 70-88): depict the names of the remaining three services, and the configuration details of these services are like the ones discussed in the *CA-localMun*. Hence, these services are not explained in detail again.

This section discussed all the four certificate authority services used by the proposed solution to issue the digital certificates of all the resources that belong to a specific organisation. Therefore, the following section focuses on the services that seek to add a new block of transactions to the Blockchain network, which is the *ordering node services*.

B.3.1.3.2 Ordering node services

This section focuses on the configuration details of all the services that seek to add data (which is a new block of transactions) to the ShareTendPro network. As indicated earlier on, the four ordering nodes (represented by node 1 to node 4 in Figure B.5) are responsible for adding new data to the Blockchain network, hence, this section explores the configuration details of the services used by these four nodes. In other words, Figure B.15 depicts the services used to create the following virtual nodes (also known as ordering nodes) namely:

orderer1.ordererOrg.workplace (represented by line 91), *orderer2.ordererOrg.workplace* (represented by line 125), *orderer3.ordererOrg.workplace* (represented by line 159), and *orderer4.ordererOrg.workplace* (represented by line 193). A detailed discussion of the ordering node services configuration as shown in Figure B.15 follows.

```

91 | orderer1.ordererOrg.workplace:
92 |   container_name: orderer1.ordererOrg.workplace
93 |   image: hyperledger/fabric-orderer:2.1
94 |   dns_search: .
95 |   environment:
96 |     - ORDERER_GENERAL_LOGLEVEL=info
97 |     - FABRIC_LOGGING_SPEC=INFO
98 |     - ORDERER_GENERAL_LISTENADDRESS=0.0.0.0
99 |     - ORDERER_GENERAL_GENESISMETHOD=file
100 |    - ORDERER_GENERAL_GENESISFILE=/var/hyperledger/orderer/genesis.block
101 |    - ORDERER_GENERAL_LOCALMSPID=OrdererOrgMSP
102 |    - ORDERER_GENERAL_LOCALMSPDIR=/var/hyperledger/orderer/msp
103 |    - ORDERER_GENERAL_TLS_ENABLED=true
104 |    - ORDERER_GENERAL_TLS_PRIVATEKEY=/var/hyperledger/orderer/tls/server.key
105 |    - ORDERER_GENERAL_TLS_CERTIFICATE=/var/hyperledger/orderer/tls/server.crt
106 |    - ORDERER_GENERAL_TLS_ROOTCAS=[/var/hyperledger/orderer/tls/ca.crt]
107 |    - ORDERER_KAFKA_VERBOSE=true
108 |    - ORDERER_GENERAL_CLUSTER_CLIENTCERTIFICATE=/var/hyperledger/orderer/tls/server.crt
109 |    - ORDERER_GENERAL_CLUSTER_CLIENTPRIVATEKEY=/var/hyperledger/orderer/tls/server.key
110 |    - ORDERER_GENERAL_CLUSTER_ROOTCAS=[/var/hyperledger/orderer/tls/ca.crt]
111 |    - ORDERER_METRICS_PROVIDER=prometheus
112 |    - ORDERER_OPERATIONS_LISTENADDRESS=0.0.0.0:8443
113 |    - ORDERER_GENERAL_LISTENPORT=7050
114 |   working_dir: /opt/gopath/src/github.com/hyperledger/fabric/orderers
115 |   command: orderer
116 |   ports:
117 |     - 7050:7050
118 |     - 8443:8443
119 |   volumes:
120 |     - ./channel/channel-artifacts/genesis.block:/var/hyperledger/orderer/genesis.block
121 |     - ./channel/crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer1.ordererOrg.workplace/msp:/var/hyperledger/orderer/msp
122 |     - ./channel/crypto-config/ordererOrganizations/ordererOrg.workplace/orderers/orderer1.ordererOrg.workplace/tls:/var/hyperledger/orderer/tls
123 |   networks:
124 |     - test
125 | orderer2.ordererOrg.workplace:
159 | orderer3.ordererOrg.workplace:
193 | orderer4.ordererOrg.workplace:
229 | couchdb0:

```

Figure B.15 Ordering node services

The following item explores the configuration details of the four ordering node services highlighted within Figure B.15, as follows:

- **Orderer1.ordererOrg.workplace:** depicts the configuration details of the first ordering node service, represented by lines 91-124, as seen in Figure B.15. The docker container used by this service is *orderer1.ordererOrg.workplace* (as seen in line 92), while the docker image used by this service is *fabric_orderer:2.1* (as seen in line 93). Lines 116-118 represent the exposed ports used by this service, which are ports 7050 and 8443. This service use port 7050 as a general listen port (as seen in line 113), while port 8443 is used as an operational listen port (as seen in line 112). Lines 121 and 122 represents the references of the crypto materials used by this service, while line 120 references the genesis block created in the previous section or stored within the channel-artifacts folder. Lines 95-113 depicts the environment variables used by this service and some of these services are:
 - a) Line 98 represents the general listen address used by this service, which is 0.0.0.0.

- b) Line 99 and 100 represents the variables used by the genesis block, whereby line 99 set a method that would accept a genesis file referenced by line 100, which is *genesis.block*. The *genesis.block* file referenced within this variable is the channel artifact generated in Figure B.12.
- c) Line 101 and 102 depicts the variables used to set the details of the channel member. Line 101 represents the *ID* used to identify the channel member, which is *OrdererOrgMSP*, while line 102 representing the *MSP* crypto materials used by the channel member.
- d) Line 103-106 represents the variables used to secure the communication channel whenever this service interacts with the Blockchain network.
- e) Line 107-110 depicts the variables used by the Raft cluster.
- **Orderer2.ordererOrg.workplace**: depicts the name of the second ordering node service, represented by lines 125-158. The configuration details of this service are similar to the ones discussed within the *orderer1.ordererOrg.workplace* – hence, lines 126-158 are hidden (collapsed) to save space on the page. This notion also applies to the configuration details of the following services: *orderer3.ordererOrg.workplace* (represented by lines 159-192), and *orderer4.ordererOrg.workplace* (represented by lines 193-227). These services are not explained in detail again, hence, it is hidden too.

This section discussed all the services that have the capabilities of adding new blocks of transactions to the ShareTendPro network. Therefore, the section to follow focuses on the services rendered by the database used by the proposed solution to store information to the Blockchain network, which is the CouchDB services.

B.3.1.3.3 CouchDB services

This section focuses on the configuration details of the services that seek to store information or data within the ledger or Blockchain network. As indicated in Chapter 3, Blockchain technology (BCT) makes use of a distributed ledger system (DLS) to store its information, which consists of two components namely: *world-state* and *Blockchain*. A *world-state* component is used to store information related to the values presented in the new state as it transits from one state to the next, while a *Blockchain* component is used to store information related to the transition logs generated by the data that has resulted in a new state. Additionally, all peer nodes within the Blockchain network contain a ledger, hence, it can be viewed as a shared ledger as discussed in Chapter 3. However, this section focuses on the mechanism or

services that store information within the *world-state*, and the proposed solution makes use of the CouchDB database to store the project information, which is contained within the *world-state*. A CouchDB is an open-source NoSQL database that seeks to collect and store data using the JSON format [131]. Therefore, Figure B.16 depicts all the CouchDB services used by the proposed solution. A more detailed discussion of the CouchDB services configuration as shown in Figure B.16 that follows next.

```

228 couchdb0:
229   container_name: couchdb0
230   image: hyperledger/fabric-couchdb
231   environment:
232     - COUCHDB_USER=
233     - COUCHDB_PASSWORD=
234   ports:
235     - 5984:5984
236   networks:
237     - test
238 couchdb1:
239 couchdb2:
240 couchdb3:
241 couchdb4:
242 couchdb5:
243 couchdb6:
244 couchdb7:
245 peer0.localMun.workplace:

```

Figure B.16 Couchdb services

The following items explore the configuration details of these CouchDB services in detail:

- **CouchDB0** depicts the configuration details of the first instance of the CouchDB database or service, as seen in lines 228-237 of Figure B.16. The docker container name of this service is *couchdb0*, while the docker image used by this service is *fabric-CouchDB*, as seen in lines 229 and 230 respectively. Lines 231-233 represent the environmental variables used by this service, which are the *username* and *password* fields. These variables can also be used to secure the information stored within the ledger (*world-state*), however, in our case these variables contain null². The exposed port used by the service is 5984, as shown in line 235.
- **CouchDB1** depicts the configuration details of the second instance of the CouchDB database or service, as shown in lines 238-250 of Figure B.16. The configuration details highlighted within this service are like the ones discussed in *couchDB0*. Hence, the configuration details of *couchDB1* are hidden or not discussed in detail again.

² In reality, the username and password fields should contain real credentials, however, since the implementation of the proposed solution is a prototype, the security of the information stored within the CouchDB services was omitted or not implemented, but in a fully-functional environment, it should be implemented.

Therefore, this notion can also be applied to other CouchDB instances or services such as *couchDB2* to *couchDB7* (represented by lines 249-314).

This section discussed all the services rendered by the CouchDB database. Therefore, the following section focuses on the services that depend on these CouchDB services, which are the peer node services.

B.3.1.3.4 Peer node services

This section focuses on the configuration details of all the services rendered by the peer nodes within the Blockchain network. A general overview of all the peer node services is first provided, followed by a more detailed discussion. As indicated in the previous chapters, the ShareTendPro consists of four organisations (Local Municipality, District Municipality, Investigator's Firm, and Auditor's Firm) and each of these organisations consists of two peer nodes – hence, the proposed solution consists of eight peer nodes, which results in eight CouchDB instances (as shown in Figure B.16). Therefore, Figure B.17 represents the configuration details of all the eight peer node services namely *peer0.localMun.workplace* (represented by lines 315-352), *peer1.localMun.workplace* (represented by lines 353-394), *peer0.districtMun.workplace* (represented by lines 395-435), *peer1.districtMun.workplace* (represented by lines 436-477), *peer0.investigatorFir.workplace* (represented by lines 478-519), *peer1.investigatorFir.workplace* (represented by lines 520-561), *peer0.auditorFir.workplace* (represented by lines 562-603), and *peer1.auditorFir.workplace* (represented by lines 604-645). A more detailed discussion of the peer node services configuration as shown in Figure B.17 that follows.


```

315 peer0.localMun.workplace:
316   container_name: peer0.localMun.workplace
317   extends:
318     file: base.yaml
319     service: peer-base
320   environment:
321     - FABRIC_LOGGING_SPEC=DEBUG
322     - ORDERER_GENERAL_LOGLEVEL=debug
323     - CORE_PEER_LOCALMSPID=LocalMunMSP
324     - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=artifacts_test
325     - CORE_PEER_ID=peer0.localMun.workplace
326     - CORE_PEER_ADDRESS=peer0.localMun.workplace:7051
327     - CORE_PEER_LISTENADDRESS=0.0.0.0:7051
328     - CORE_PEER_CHAINCODEADDRESS=peer0.localMun.workplace:7052
329     - CORE_PEER_CHAINCODELISTENADDRESS=0.0.0.0:7052
330     - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer0.localMun.workplace:7051
331     - CORE_PEER_GOSSIP_BOOTSTRAP=peer1.localMun.workplace:8051
332     - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
333     - CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=couchdb0:5984
334     - CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=
335     - CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=
336     - CORE_METRICS_PROVIDER=prometheus
337     - CORE_PEER_TLS_ENABLED=true
338     - CORE_PEER_TLS_CERT_FILE=/etc/hyperledger/crypto/peer/tls/server.crt
339     - CORE_PEER_TLS_KEY_FILE=/etc/hyperledger/crypto/peer/tls/server.key
340     - CORE_PEER_TLS_ROOTCERT_FILE=/etc/hyperledger/crypto/peer/tls/ca.crt
341     - CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/crypto/peer/msp
342   depends_on:
343     - couchdb0
344   ports:
345     - 7051:7051
346   volumes:
347     - ./channel/crypto-config/peerOrganizations/localMun.workplace/peers/peer0.localMun.workplace/msp:/etc/hyperledger/crypto/peer/msp
348     - ./channel/crypto-config/peerOrganizations/localMun.workplace/peers/peer0.localMun.workplace/tls:/etc/hyperledger/crypto/peer/tls
349     - /var/run:/host/var/run/
350     - ./channel/channel-artifacts:/etc/hyperledger/channel/
351   networks:
352     - test
353 peer1.localMun.workplace:
354 peer0.districtMun.workplace:
355 peer1.districtMun.workplace:
356 peer0.investigatorFir.workplace:
357 peer1.investigatorFir.workplace:
358 peer0.auditorFir.workplace:
359 peer1.auditorFir.workplace:
646

```

Figure B.17 Peer node services

The following items explore the configuration details of all the peer node services highlighted within Figure B.17 in more detail:

- **Peer0.localMun.workplace** depicts the first service rendered by one of the peer nodes that belongs to the Local Municipality, as shown in line 315 of Figure B.17. The docker container name of this service is *peer0.localMun.workplace*, as seen in line 316. Lines 317-319 explore some of the default configuration details contained within the file called *base.yaml* (as shown in line 318). Lines 342-343 depict the elements that seek to configure a CouchDB service used by this service, which is *couchdb0* as shown in line 343. Line 345 represents the exposed port used by this service, which is port 7051, while lines 347-350 reference the crypto materials used by this service, as well as the channel artifacts. Lines 320-341 depict the environmental variables used by this service as follows:
 - a. Line 323 seeks to configure the channel member used by this service, which is *LocalMunMSP*.
 - b. Line 324 depicts a variable used to set up the name of the virtual network used by the proposed solution, which is *artifacts_test*.
 - c. Lines 325-331 represent the variables used by the peer nodes, whereby the peer node *ID* used is represented by line 325. Lines 326 and 327 depict the addresses

used by this service to interact with the Blockchain network. Lines 328 and 329 represent the addresses used by this service to interact with the chaincode (which is discussed in the section to follow), while lines 330 and 331 represent the addresses used by this service to share project information with other nodes that belong to its domain name.

- d. Lines 332-335 represent the variables used to set up the CouchDB service used by this node as discussed in the previous section.
 - e. Lines 337-340 depict the variables used to secure the communication channel whenever this service interacts with the Blockchain network.
- **Peer1.localMun.workplace** represents the second service rendered by one of the peer nodes that belongs to the Local Municipality, as shown in line 353 of Figure B.17. However, all the configuration details contained within this service are similar to the ones discussed in *peer0.localMun.workplace*. Hence, the configuration details of *peer1.localMun.workplace* service are hidden as part of trying to avoid repeating some of the concepts, which implies that there are not explained in detail again. This notion can also be applied to the following remaining peer node services *peer0.districtMun.workplace* (represented by lines 395-435), *peer1.districtMun.workplace* (represented by lines 436-477), *peer0.investigatorFir.workplace* (represented by lines 478-519), *peer1.investigatorFir.workplace* (represented by lines 520-561), *peer0.auditorFir.workplace* (represented by lines 562-603), and *peer1.auditorFir.workplace* (represented by lines 604-645).

This section discussed all the necessary information required to generate a network topology of the proposed solution, which is the ShareTendPro model. All the configuration details related to the three important files (*crypto-config.yaml* file, *configtx.yaml* file, and *docker-compose.yaml* file) have been explored in detail. Therefore, the following section focuses on the development of the rules (also known as chaincode) that seek to govern the Blockchain network. The following section explores the chaincode (also known as a smart-contract) development that allows various organisations (i.e. Local Municipality, District Municipality, Investigator's Firm, and Auditor's Firm) to interact with the Blockchain network of the proposed solution.

B.3.2 Chaincode development

This section explores all the necessary information required to implement the chaincode that would be used by the proposed solution to govern the Blockchain network. As indicated in Figure B.2, the Go tool is used to implement the chaincode used by the proposed solution. Therefore, Figure B.18 depicts the chaincode development as the focus area, which is also viewed as the last step of the ShareTendPro development process. The chaincode development is divided into two main categories namely *tender structure* and *invoke functions*, as shown in Figure B.18. The *tender structure* category focuses on the data structure used to either collect or store information within the Blockchain network, while the *invoke functions* category focuses on the mechanisms used by various organisations (i.e., Local Municipality, District Municipality, Investigator’s Firm, and Auditor’s Firm) to interact with the ShareTendPro network. Additionally, the *invoke functions* category is classified into three functions namely *InitLedger*, *CreateTender*, and *QueryTender*, as shown in Figure B.18.

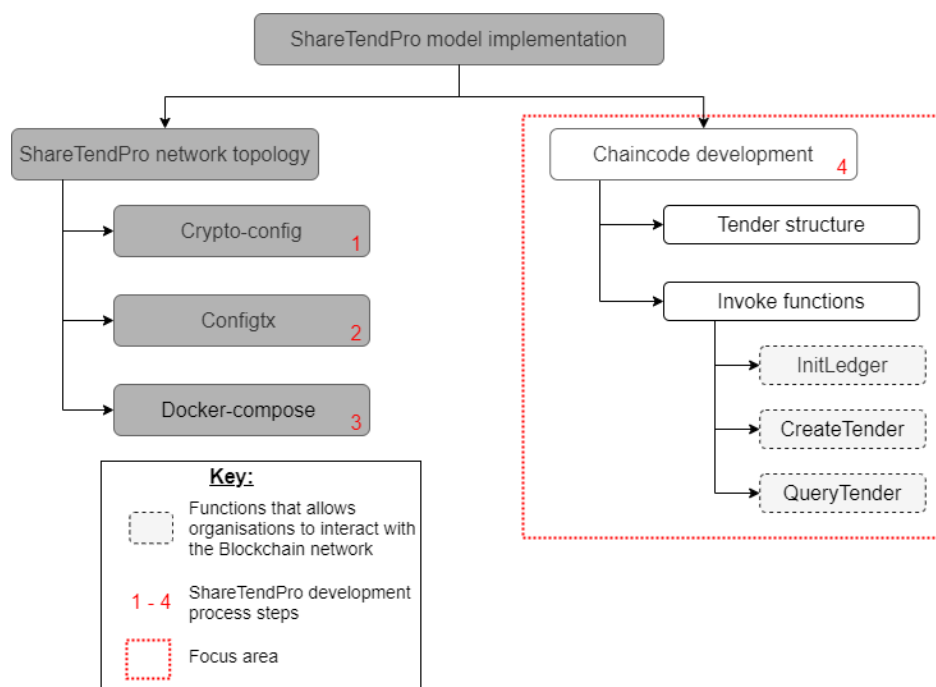


Figure B.18 Chaincode development as the focus area

The following sections explore the details contained within the *tender structure* category, followed by the details contained within the *invoke functions* category.

B.3.2.1 Tender structure

Figure B.19 represents the data structure used by the chaincode to collect project information, which is the *struct* data structure. In the Go programming language, a *struct* can be viewed as a collection of data fields grouped together to form records [132]. Therefore, Figure B.19 depicts a *struct* called *Tender* with the following fields: *TenderOwner*, *ProjectNum*, *Reports*, and *Supplier*. All these fields are of the data type string, as shown in Figure B.19. Additionally, all the data contained within each of these fields must be represented in a JSON format. A more detailed discussion of the *tender structure* as shown in Figure B.19 follows below.

```

17 // Tender : Define the tender structure, wi
18 type Tender struct {
19     TenderOwner string `json:"tenderOwner"`
20     ProjectNum   string `json:"projectNum"`
21     Reports     string `json:"reports"`
22     Supplier    string `json:"supplier"`
23 }
24

```

Figure B.19 Tender structure

The following items seek to explore the information that should be stored within each of the fields contained within the *Tender struct* as shown in Figure B.19, as soon as such a Tender object is instantiated:

- **TenderOwner:** this field would contain information related to the municipality that has issued a particular tendering project.
- **ProjectNum:** this field would contain the information related to the project or tender number assigned to that tendering project that has been issued by a specific municipality. The information stored within this field is unique because it seeks to identify a specific project within that municipality or within the Blockchain network.
- **Reports:** this field would contain information related to the project reports that need to be shared within the Blockchain network.
- **Supplier:** this field would contain information related to the contractor or supplier that has been awarded a contract or tender to render a service required by the municipality that has issued a tender project.

This section discussed the basic structure used by the proposed solution to collect project information. One of the main objectives of the proposed solution is to demonstrate how BCT might be used to share project information, hence, the basic *struct* is used to portray that narrative. Therefore, the following section focuses on the details contained within the functions

that use the *Tender struct* to interact within the Blockchain network, as the *invoke functions* category.

B.3.2.2 Invoke functions

Figure B.20 depicts an overview of the invoke functions (*initLedger* function, *createTender* function, and *queryTender* function) used by the various organisations to interact within the ShareTendPro network. As indicated in Figure B.20, the *initLedger* function is invoked using the statements represented by lines 42-43. The *createTender* function is invoked using the statements represented by lines 44-45, while the *queryTender* function is invoked using the statements represented by lines 40-41. All these functions make use of special application programming interfaces (APIs) called *shim* to interact with the ShareTendPro network. The shim APIs can be viewed as a special library designed for the chaincode or HLF framework that might be used to interact with the data or information stored within the Blockchain network [133]. Some of this information include the current state of the Blockchain network and the transactions context, which is the detailed information that constitutes to the current state of the Blockchain network.

```

33 func (s *SmartContract) Invoke(APIStub shim.ChaincodeStubInterface) sc.Response {
34
35     function, args := APIStub.GetFunctionAndParameters()
36
37     logger.Infof("Function name is: %d", function)
38     logger.Infof("Args length is : %d", len(args))
39
40     if function == "queryTender" {
41         return s.queryTender(APIStub, args)
42     } else if function == "initLedger" {
43         return s.initLedger(APIStub)
44     } else if function == "createTender" {
45         return s.createTender(APIStub, args)
46     }
47     return shim.Error("Invalid Smart Contract function name.")
48 }

```

Figure B.20 Invoke functions overview

The following subsections seek to explore the details contained within each of the functions highlighted in Figure B.20, starting with the *initLedger* function, followed by the *createTender* function in the second subsection, then ended with the *queryTender* function in the last section. The *initLedger* subsection focuses on the information that seeks to initialise the ledger within the Blockchain network. The *createTender* focuses on the information that seeks to create or add project information within the Blockchain network. The *queryTender* focuses on the information that might be used to access or retrieve project information stored within the Blockchain network.

B.3.2.2.1 InitLedger

Figure B.21 depicts a function that seeks to initialise some of the data or project information. Lines 61-65 focus on collecting data as a group of records using the *struct* data structure, while lines 67-74 focus on storing these records of data to the Blockchain network. A more detailed discussion of the `initLedger` function as shown in Figure B.21 follows

```

60 ~ func (s *SmartContract) initLedger(APIStub shim.ChaincodeStubInterface) sc.Response {
61 ~     tenders := []Tender{
62 ~         Tender{TenderOwner: "Tshwane Municipality", ProjectNum: "p11112", Reports: "project started", Supplier: "MS Trade"},
63 ~         Tender{TenderOwner: "Polokwane Municipality", ProjectNum: "p22221", Reports: "10% project completed", Supplier: "MTN"},
64 ~         Tender{TenderOwner: "Johannesburg Municipality", ProjectNum: "p33331", Reports: "IT project started", Supplier: "IBM"}
65 ~     }
66 ~
67 ~     i := 0
68 ~     for i < len(tenders) {
69 ~         tenderAsBytes, _ := json.Marshal(tenders[i])
70 ~         APIStub.PutState("Tender"+strconv.Itoa(i), tenderAsBytes)
71 ~         i = i + 1
72 ~     }
73 ~
74 ~     return shim.Success(nil)
75 ~ }
76 ~

```

Figure B.21 IniLedger function

The following items explore the details contained in one of the records of data presented by line 62 of Figure B.21:

- **TenderOwner:** as indicated in the previous section, this field contains information related to the municipality that issued the tendering project. In this case, the municipality that has issued the tendering project is called *Tshwane Municipality*.
- **ProjectNum:** as indicated in the previous section, this field contains information related to the project number assigned to that tendering project. In this case, the project number assigned to the tendering project issued by Tshwane Municipality is *p11112*.
- **Reports:** as indicated in the previous section, this field contains information related to the project report of the tendering project issued by Tshwane Municipality. In this case, the project report assigned to this tendering project is “*project started*”.
- **Supplier:** as indicated in the previous section, this field contains information related to the contractor that was awarded the tendering project to render that service to Tshwane Municipality. In this case, the supplier assigned to the tendering project is called “*MS Trade*”.

Lines 67-72 depict a “*For loop*” that might be used to collect all the records of data represented by a slice called *tenders* (as shown in line 61). In the Go programming language, a slice can be viewed as a key data type designed with a more powerful interface to sequences because it is dynamically sized and flexible than an array [134]. Line 69 depicts a statement that seeks to convert the data stored within a slice called *tenders* into a JSON format, while line 70 represents

a statement that seeks to add the data to the Blockchain network. Thereafter, line 74 represents a statement that seeks to return a *shim* result generated after executing this function or successfully adding the data to the Blockchain network.

The following section explores the details of a function called *createTender*, which is a function that adds project information to the Blockchain network.

B.3.2.2.2 CreateTender

Figure B.22 depicts a function that might be used by various organisations to add project information to the Blockchain network. Line 79 depicts a statement that checks whether the arguments passed to this function are five and if there are not five then an error message represented by line 80 is going to be executed. As indicated in line 77, this function is capable of taking as many arguments as possible because these arguments are represented in a form of an array, i.e. `args []string`. Hence, the maximum number of arguments required by this function to add a new record of transactions is five, whereby the first argument represented by index 0 should contain a unique value that can be assigned to a particular record, while the other four arguments represented by index 1 – 4 should contain the data related to a specific tendering project or the data stored within the *Tender* struct. Line 83 represents a variable called *tender* that might be used to collect project information using a struct data structure called *Tender*. The first argument of the struct *Tender* is assigned to a field called *TenderOwner*, the second argument is assigned to a field called *ProjectNum*, the third argument is assigned to a field called *Reports*, while the fourth argument is assigned to a field called *Supplier*. Line 85 represents a statement that seeks to convert the data stored within the *tender* variable to a JSON format, while line 86 depicts a statement that seeks to store that data to the Blockchain network. Thereafter, line 88 represents a statement that seeks to return a *shim* result generated after executing this function or successfully adding the data to the Blockchain network.

```

77 ~ func (s *SmartContract) createTender(APIStub shim.ChaincodeStubInterface, args []string) sc.Response {
78
79 ~   if len(args) != 5 {
80     return shim.Error("Incorrect number of arguments. Expecting 5")
81   }
82
83   var tender = Tender{TenderOwner: args[1], ProjectNum: args[2], Reports: args[3], Supplier: args[4]}
84
85   tenderAsBytes, _ := json.Marshal(tender)
86   APIStub.PutState(args[0], tenderAsBytes)
87
88   return shim.Success(tenderAsBytes)
89 }

```

Figure B.22 CreateTender function

The following section explores the details contained within a function called *queryTender*, which is a function that seeks to access or retrieve project information from the Blockchain network.

B.3.2.2.3 QueryTender

Figure B.23 depicts a function that might be used by various organisations to access or retrieve project information from the Blockchain network. Line 52 represents a statement that checks whether the number of arguments passed is one and if it is not 1 then an error message represented by line 53 is going to be executed. The main reason for trying to check whether one argument has been passed to this function is that the Shim API used within this function takes more than one argument as seen in line 50, whereby the representation of the arguments passed are in a form of an array (i.e. `args []string`). However, the argument passed within this function should match a specific record within the Blockchain network, for it to access data or tendering project information. Therefore, line 56 represents a statement that seeks to retrieve the data from the Blockchain network, while line 57 depicts a statement that seeks to return a shim result generated after executing this function or successfully retrieving the data from the Blockchain network.

```

50 func (s *SmartContract) queryTender(APIstub shim.ChaincodeStubInterface, args []string) sc.Response {
51
52     if len(args) != 1 {
53         return shim.Error("Incorrect number of arguments. Expecting 1")
54     }
55
56     tenderAsBytes, _ := APIstub.GetState(args[0])
57     return shim.Success(tenderAsBytes)
58 }

```

Figure B.23 QueryTender function

B.4 Summary

This chapter explored the implementation of the proposed solution, which includes the various tools used to develop the prototype and ShareTendPro model implementation. The tools used to develop the prototype are divided into two requirements namely hardware and software requirements, while the implementation of the ShareTendPro model is divided into two namely the ShareTendPro network topology and chaincode development. Additionally, the ShareTendPro network topology focused on the implementation of the Blockchain network, while the chaincode development focused on the rules that govern the Blockchain network.

The following chapter delves into the results generated from the proposed solution as part of trying to present how the proposed solution work in general.

C. Appendix C: ShareTendPro network results

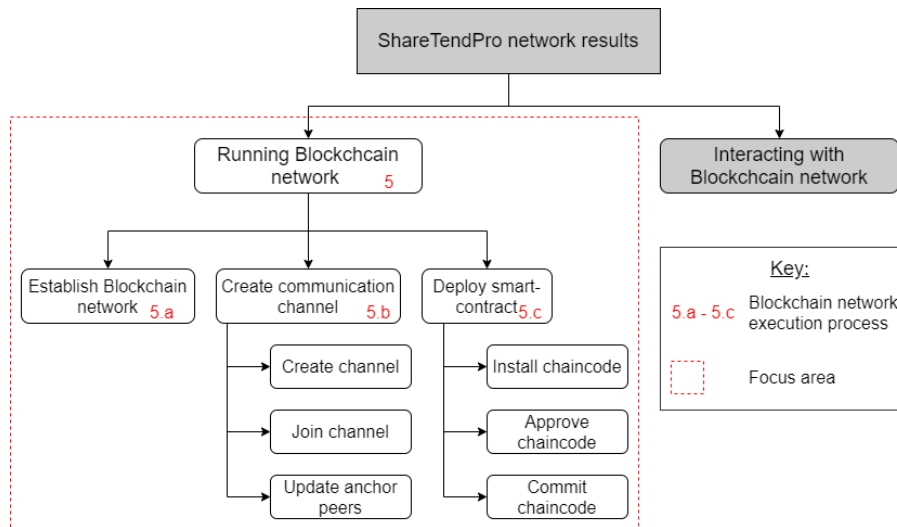


Figure C.1 ShareTendPro network results overview

C.1 Establish Blockchain network

This section focuses on the results generated after executing the configuration file called *docker-compose.yaml* using the following command line “*docker-compose up -d*”. Therefore, Figure C.2 depicts the results displayed within the command prompt after executing this command line. Label CL of Figure C.2 depicts the command line used to execute the *docker-compose.yaml* file, while label R represents the results generated after executing the *docker-compose.yaml* file. Note that all the results are represented using the docker container name or hostname name of the services discussed in the previous chapter and Appendix B. Additionally, all the results represented in Figure C.2 sought to generate the virtual nodes that form part of the Blockchain or virtual network used by the ShareTendPro network

```

base.yaml █████ docker-compose.yaml █████
pardon@pardon-VirtualBox:~/Documents/mscProject/artifacts$ docker-compose up -d
Creating network "artifacts_test" with the default driver
Creating ca.investigatorFir.workplace ...
Creating ca.localMun.workplace ...
Creating couchdb1 ...
Creating couchdb3 ...
Creating orderer4.ordererOrg.workplace ...
Creating ca.localMun.workplace
Creating orderer3.ordererOrg.workplace ...
Creating couchdb7 ...
Creating ca.investigatorFir.workplace
Creating couchdb1
Creating couchdb3
Creating orderer3.ordererOrg.workplace
Creating orderer1.ordererOrg.workplace ...
Creating couchdb0 ...
Creating orderer4.ordererOrg.workplace
Creating couchdb7
Creating ca.auditorFir.workplace ...
Creating orderer1.ordererOrg.workplace
Creating couchdb4 ...
Creating couchdb0
Creating orderer2.ordererOrg.workplace ...
Creating couchdb2 ...
Creating ca.auditorFir.workplace
Creating ca.districtMun.workplace ...
Creating couchdb4
Creating couchdb6 ...
Creating orderer2.ordererOrg.workplace
Creating couchdb2
Creating couchdb5 ...
Creating ca.districtMun.workplace
Creating couchdb5
Creating couchdb1 ... done
Creating peer1.localMun.workplace ...
Creating couchdb3 ... done
Creating peer1.districtMun.workplace ...
Creating couchdb2 ... done
Creating peer0.districtMun.workplace ...
Creating couchdb0 ... done
Creating peer0.localMun.workplace ...
Creating couchdb5 ... done
Creating peer1.investigatorFir.workplace ...
Creating couchdb4 ... done
Creating peer0.investigatorFir.workplace ...
Creating couchdb6 ... done
Creating peer0.auditorFir.workplace ...
Creating couchdb7 ... done
Creating peer1.auditorFir.workplace ...
Creating peer0.investigatorFir.workplace ... done

```

Figure C.2 Docker-compose services

As indicated in the previous section, once the Blockchain network is established, a process of creating a communication channel is activated. Therefore, the following section focuses on the results generated by the process that seeks to create the communication channel used by various nodes to secretly share project information. Note that this communication channel is created within the Blockchain network.

C.2 Create communication channel

This section focuses on the results generated during the process of creating a private communication channel within the Blockchain network. However, the results contained within this section are classified into three subsections namely: *create channel*, *join channel*, and *update anchor peers*, as shown in Figure C.2. The first subsection, which is *create channel* focuses on the results generated after executing a command line that seeks to create the communication channel within the Blockchain network. The second subsection, which is *join*

channel focuses on the results generated after executing the command lines that seek to join the peer nodes (virtual nodes created in Figure C.2) of various organisations to the communication channel created using the *create channel* subsection. Lastly, the third subsection, which is *update anchor peers* subsection focuses on the results generated after executing the command lines that seek to update the roles of some of the peer nodes within the Blockchain network. As illustrated in the previous chapter, one of the roles of the anchor peer node is to discover other peer nodes that fall outside their domain name.

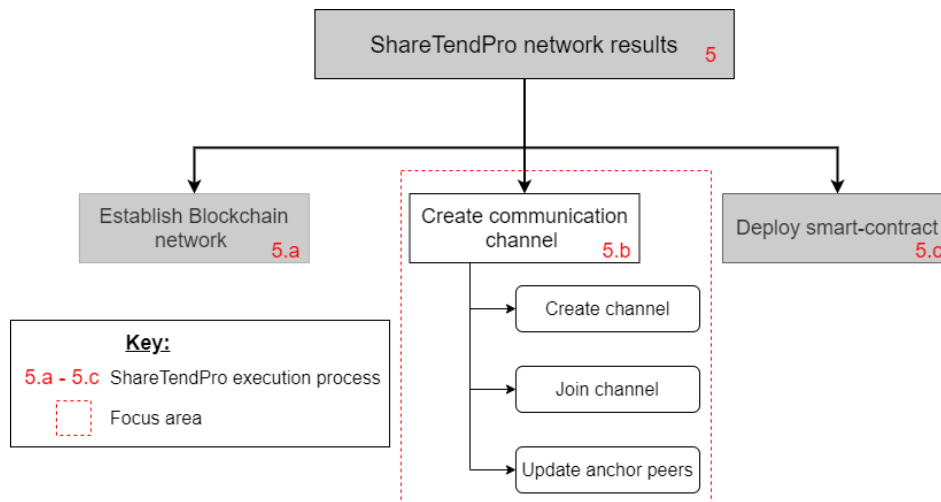


Figure C.3 Create communication channel as the focus area

C.2.1 Create channel

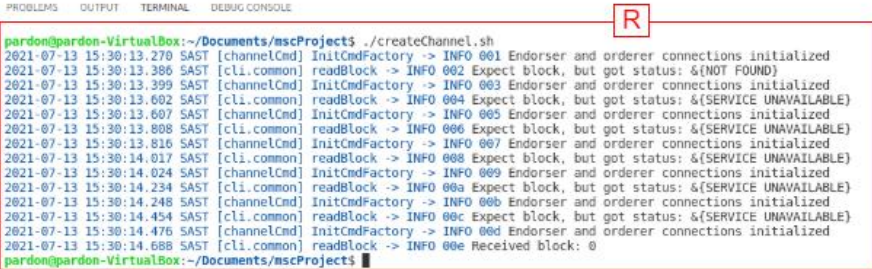
This section details the results generated during the process of creating the communication channel within the Blockchain network. Therefore, lines 55-60 of Figure C.4 depict a function called *setGlobalsForPeer0AuditorFirm()* that contains the variables for *peer0* of the Auditor’s Firm. Lines 68-73 represent a function called *createChannel* that contains the command line that seeks to create the communication channel within the Blockchain network using the variables contained within the *setGlobalsForPeer0AuditorFirm()* function, when calling this function in line 69. Lines 70-72 represent the command line used by the *createChannel()* function to create the communication channel. As indicated in line 71, this command line uses a *\${CHANNEL_NAME}.tx* artifact contained within the *channel-artifact* folder, and the output or final results generated after executing this command line are stored in a file called *\${CHANNEL_NAME}.block*, where *\${CHANNEL_NAME}* is a string variable assigned as “*mychannel*”. Label R of Figure C.4 represents the results displayed within the command

prompt after executing the command line contained within the *createChannel* function using line 74.

```

55  setGlobalsForPeer0AuditorFirm(){
56      export CORE_PEER_LOCALMSPID="AuditorFirmSP"
57      export CORE_PEER_TLS_ROOTCERT_FILE=$PEER0_AuditorFirm_CA
58      export CORE_PEER_MSPCONFIGPATH=${PWD}/artifacts/channel/crypto-config/peerOrganizations/auditorFirm.workplace/users/Admin@auditorFirm.workplace/msp
59      export CORE_PEER_ADDRESS=localhost:13051
60  }
61  > setGlobalsForPeer1AuditorFirm(){--
62  }
63  createChannel(){
64      setGlobalsForPeer0AuditorFirm
65      peer channel create -o localhost:7050 -c $CHANNEL_NAME --ordererTLShostnameOverride orderer1.ordererOrg.workplace \
66      -f ./artifacts/channel/channel-artifacts/${CHANNEL_NAME}.tx --outputBlock ./channel-artifacts/${CHANNEL_NAME}.block \
67      --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA
68  }
69  createChannel

```



```

pardon@pardon-VirtualBox:~/Documents/mscProject$ ./createChannel.sh
2021-07-13 15:30:13.270 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-07-13 15:30:13.386 SAST [cli.common] readBlock -> INFO 002 Expect block, but got status: &(NOT FOUND)
2021-07-13 15:30:13.399 SAST [channelCmd] InitCmdFactory -> INFO 003 Endorser and orderer connections initialized
2021-07-13 15:30:13.602 SAST [cli.common] readBlock -> INFO 004 Expect block, but got status: &(SERVICE UNAVAILABLE)
2021-07-13 15:30:13.607 SAST [channelCmd] InitCmdFactory -> INFO 005 Endorser and orderer connections initialized
2021-07-13 15:30:13.808 SAST [cli.common] readBlock -> INFO 006 Expect block, but got status: &(SERVICE UNAVAILABLE)
2021-07-13 15:30:13.816 SAST [channelCmd] InitCmdFactory -> INFO 007 Endorser and orderer connections initialized
2021-07-13 15:30:14.017 SAST [cli.common] readBlock -> INFO 008 Expect block, but got status: &(SERVICE UNAVAILABLE)
2021-07-13 15:30:14.024 SAST [channelCmd] InitCmdFactory -> INFO 009 Endorser and orderer connections initialized
2021-07-13 15:30:14.234 SAST [cli.common] readBlock -> INFO 00a Expect block, but got status: &(SERVICE UNAVAILABLE)
2021-07-13 15:30:14.248 SAST [channelCmd] InitCmdFactory -> INFO 00b Endorser and orderer connections initialized
2021-07-13 15:30:14.454 SAST [cli.common] readBlock -> INFO 00c Expect block, but got status: &(SERVICE UNAVAILABLE)
2021-07-13 15:30:14.476 SAST [channelCmd] InitCmdFactory -> INFO 00d Endorser and orderer connections initialized
2021-07-13 15:30:14.688 SAST [cli.common] readBlock -> INFO 00e Received block: 0
pardon@pardon-VirtualBox:~/Documents/mscProject$

```

Figure C.4 Create a channel within the Blockchain network

This section explored the results that were generated during the creation of the communication channel. Therefore, the following section focuses on the results generated by a process that seeks to join the peer nodes to the communication channel created in Figure C.4.

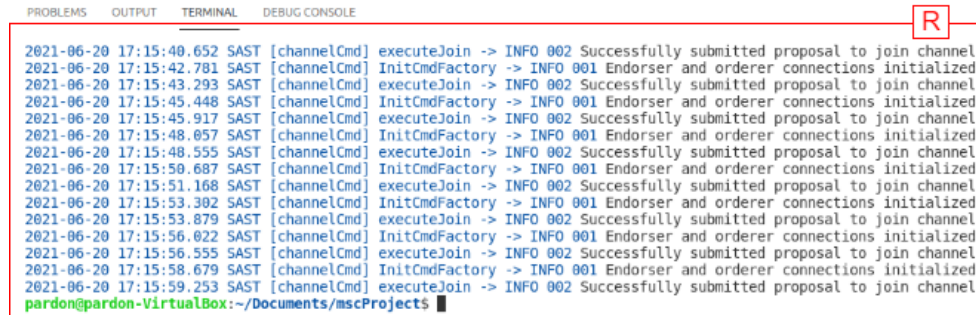
C.2.2 Join channel

This section explores the results generated by the process of joining the peer nodes to the communication channel created in the previous section. Therefore, lines 78-102 of Figure C.5 depict a function called *joinChannel* which contains the command lines that allow the peer nodes to join the channel created in Figure C.4, while label R represents the results displayed within the command prompt after executing the *joinChannel* function (represented by line 103). All these command lines make use of a binary file called *peer*. For instance, line 79 of Figure C.5 depicts a function that seeks to assign the variables of *peer0* of Local Municipality that want join the communication channel. Thereafter, line 80 represents the command line that allows *peer0* of Local Municipality to join the channel and the results of this command line are displayed in label R of Figure C.5. However, the command line represented by line 81 seeks to refresh the Blockchain network and prepares to assign the new variables of a different peer node. The notion for submitting a request to join the channel can also be applied to other command lines represented by lines 82-101 and their results are displayed in label R.

```

78 joinChannel(){
79     setGlobalsForPeer0LocalMunicipality
80     peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
81     sleep 2
82     setGlobalsForPeer1LocalMunicipality
83     peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
84     sleep 2
85     setGlobalsForPeer0DistrictMunicipality
86     peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
87     sleep 2
88     setGlobalsForPeer1DistrictMunicipality
89     peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
90     sleep 2
91     setGlobalsForPeer0AuditorFirm
92     peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
93     sleep 2
94     setGlobalsForPeer1AuditorFirm
95     peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
96     sleep 2
97     setGlobalsForPeer0InvestigatorFirm
98     peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
99     sleep 2
100    setGlobalsForPeer1InvestigatorFirm
101    peer channel join -b ./channel-artifacts/$CHANNEL_NAME.block
102 }
103 joinChannel
104

```



```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
2021-06-20 17:15:40.652 SAST [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
2021-06-20 17:15:42.781 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:15:43.293 SAST [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
2021-06-20 17:15:45.448 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:15:45.917 SAST [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
2021-06-20 17:15:48.057 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:15:48.555 SAST [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
2021-06-20 17:15:50.687 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:15:51.168 SAST [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
2021-06-20 17:15:53.302 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:15:53.879 SAST [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
2021-06-20 17:15:56.022 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:15:56.555 SAST [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
2021-06-20 17:15:58.679 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:15:59.253 SAST [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
pardon@pardon-VirtualBox:~/Documents/mscProject$ █

```

Figure C.5 Joining peer nodes to the channel

This section has explored all the results related to the command lines that seek to allow peer nodes to submit the proposal to join the communication channel within the Blockchain network. Therefore, the following section explores the results related to the command lines used to update the roles of certain peer nodes that have joined the channel to anchor peer nodes.

C.2.3 Update anchor peers

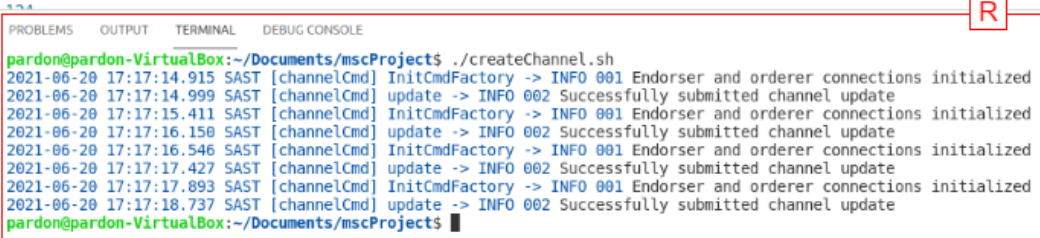
This section focuses on the results generated by updating the roles of some of the peer nodes within the Blockchain network. Therefore, lines 105-122 of Figure C.6 depict a function called *updateAnchorPeers* which contains the command lines used to update the anchor peer nodes, while label R represents the results generated after executing the function as represented by line 123. Line 106 represents a command line that seeks to assign the variables of *peer0* of the Local Municipality. Line 107-109 represents a command line used to update the role of the *peer0* of the Local Municipality to an anchor peer node. Additionally, the command line makes use of a binary file called *peer* to execute the command line, while other information used by the command is; the details of an ordering node (which is *orderer1.ordererOrg.workplace*) and the details of an artifact called *LocalMunMSPanchors.tx*. However, in Figure C.6, each variable

represented by `{CORE_PEER_LOCALMSPID}` is assigned a respective MSP ID for a specific organisation and in this case, it represents *LocalMunMSP*. This notion can be applied to other command lines that seek to update the anchor peer nodes within the Blockchain network and the results of these command lines are represented in label R of Figure C.6.

```

105 updateAnchorPeers(){
106     setGlobalsForPeer@LocalMunicipality
107     peer channel update -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
108         -c $CHANNEL_NAME -f ./artifacts/channel/channel-artifacts/${CORE_PEER_LOCALMSPID}anchors.tx \
109         --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA
110     setGlobalsForPeer@DistrictMunicipality
111     peer channel update -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
112         -c $CHANNEL_NAME -f ./artifacts/channel/channel-artifacts/${CORE_PEER_LOCALMSPID}anchors.tx \
113         --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA
114     setGlobalsForPeer@InvestigatorFirm
115     peer channel update -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
116         -c $CHANNEL_NAME -f ./artifacts/channel/channel-artifacts/${CORE_PEER_LOCALMSPID}anchors.tx \
117         --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA
118     setGlobalsForPeer@AuditorFirm
119     peer channel update -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
120         -c $CHANNEL_NAME -f ./artifacts/channel/channel-artifacts/${CORE_PEER_LOCALMSPID}anchors.tx \
121         --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA
122 }
123 updateAnchorPeers
124

```



```

pardon@pardon-VirtualBox:~/Documents/mscProject$ ./createChannel.sh
2021-06-20 17:17:14.915 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:17:14.999 SAST [channelCmd] update -> INFO 002 Successfully submitted channel update
2021-06-20 17:17:15.411 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:17:16.150 SAST [channelCmd] update -> INFO 002 Successfully submitted channel update
2021-06-20 17:17:16.546 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:17:17.427 SAST [channelCmd] update -> INFO 002 Successfully submitted channel update
2021-06-20 17:17:17.893 SAST [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2021-06-20 17:17:18.737 SAST [channelCmd] update -> INFO 002 Successfully submitted channel update
pardon@pardon-VirtualBox:~/Documents/mscProject$ █

```

Figure C.6 Updating anchor peer nodes

This section explored the results generated by the command lines that seek to create the communication channel within the Blockchain network. The following section focuses on the results of a script that seeks to deploy the smart-contract to the Blockchain network since it allows the peer nodes to interact with the Blockchain data.

C.3 Deploy smart-contract

This section focuses on the results generated by the smart-contract mechanism Figure C.7. However, the results contained within this section are classified into four categories namely: *install chaincode*, *approve chaincode*, *commit chaincode*, and *invoke chaincode*. Note that from now onwards the word “*chaincode*” would be used to refer to “*smart-contract*”. The first category, which is *install chaincode* focuses on the results generated after executing the command lines that seek to install chaincode to all the peer nodes within the Blockchain network. The second category, which is *approve chaincode* focuses on the results generated after executing the command lines that seek to approve the chaincode installed in all the peer nodes within the Blockchain network. The third category, which is *commit chaincode* focuses

on the results generated after executing the command lines that seek to commit the approved chaincode within the Blockchain network. Lastly, the fourth category, which is *invoke chaincode* focuses on the results generated after executing the command lines that allow various peer nodes to interact with the Blockchain network or data.

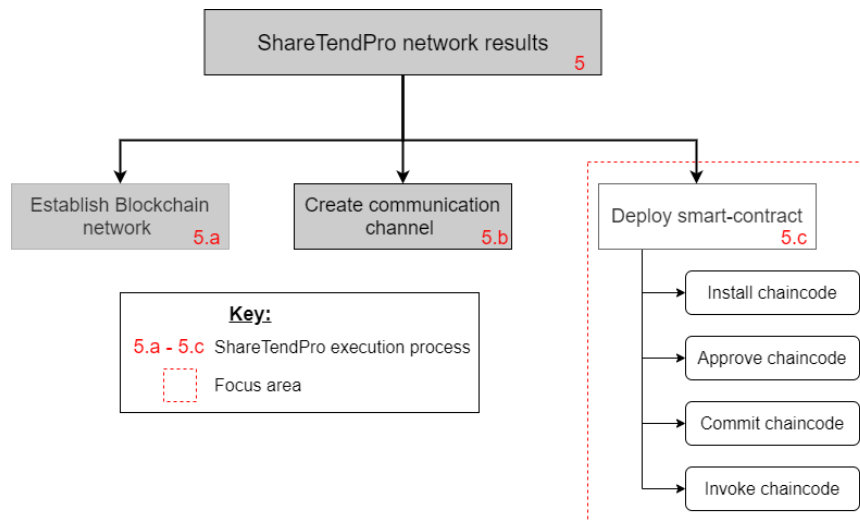


Figure C.7 Deploy smart-contract as the focus area

C.3.1 Install chaincode

This section focuses on the results generated during the process of installing the chaincode (smart-contract) to all the peer nodes that have joined the communication channel within the Blockchain network. Therefore, lines 90-115 of Figure C.8 depict a function called *installChaincode* which contains the command lines that seek to install the chaincode to all the peer nodes within the Blockchain network, while label R depicts the results generated after executing this function as presented by line 116. For instance, line 91 represents a function that seeks to assign the variables of *peer0* of Local Municipality. Line 92 represents the command line used to install the chaincode within *peer0* of Local Municipality, while line 93 aimed at separating the results of this command line from other results, as there are being displayed within the command prompt. Additionally, the command line represented by line 92 makes use of a binary file called *peer* to install the chaincode which is packaged in a file called $\${CC_NAME}.tar.gz$, whereby the $\${CC_NAME}$ is assigned to *fabtender*. Note that this notion can be applied to the remaining command lines represented by lines 94-114, and their results are also displayed in label R.

```

90  installChaincode() {
91      setGlobalsForPeer0LocalMunicipality
92      peer lifecycle chaincode install ${CC_NAME}.tar.gz
93      echo "===== Chaincode is installed on peer0.localMun.workplace ===== "
94      setGlobalsForPeer1LocalMunicipality
95      peer lifecycle chaincode install ${CC_NAME}.tar.gz
96      echo "===== Chaincode is installed on peer1.localMun.workplace ===== "
97      setGlobalsForPeer0DistrictMunicipality
98      peer lifecycle chaincode install ${CC_NAME}.tar.gz
99      echo "===== Chaincode is installed on peer0.districtMun.workplace ===== "
100     setGlobalsForPeer1DistrictMunicipality
101     peer lifecycle chaincode install ${CC_NAME}.tar.gz
102     echo "===== Chaincode is installed on peer1.districtMun.workplace ===== "
103     setGlobalsForPeer0InvestigatorFirm
104     peer lifecycle chaincode install ${CC_NAME}.tar.gz
105     echo "===== Chaincode is installed on peer0.investigatorFir.workplace ===== "
106     setGlobalsForPeer1InvestigatorFirm
107     peer lifecycle chaincode install ${CC_NAME}.tar.gz
108     echo "===== Chaincode is installed on peer1.investigatorFir.workplace ===== "
109     setGlobalsForPeer0AuditorFirm
110     peer lifecycle chaincode install ${CC_NAME}.tar.gz
111     echo "===== Chaincode is installed on peer0.auditorFir.workplace ===== "
112     setGlobalsForPeer1AuditorFirm
113     peer lifecycle chaincode install ${CC_NAME}.tar.gz
114     echo "===== Chaincode is installed on peer1.auditorFir.workplace ===== "
115 }
116 installChaincode

```

```

garden@gardon-VirtualBox:~/Documents/mscProjects $ ./deployChaincode.sh
===== Chaincode is packaged on peer0.localMun.workplace =====
2021-07-06 09:59:28.115 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 001 Installed remotely: response=<status:200 payload:"\nfabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2,022\013fabtender 1" >
===== Chaincode is installed on peer0.localMun.workplace =====
2021-07-06 09:59:32.291 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 001 Installed remotely: response=<status:200 payload:"\nfabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2,022\013fabtender 1" >
2021-07-06 09:59:32.293 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 002 Chaincode code package identifier: fabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2
===== Chaincode is installed on peer1.localMun.workplace =====
2021-07-06 09:59:34.895 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 001 Installed remotely: response=<status:200 payload:"\nfabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2,022\013fabtender 1" >
2021-07-06 09:59:34.895 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 002 Chaincode code package identifier: fabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2
===== Chaincode is installed on peer0.districtMun.workplace =====
2021-07-06 09:59:39.157 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 001 Installed remotely: response=<status:200 payload:"\nfabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2,022\013fabtender 1" >
2021-07-06 09:59:39.157 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 002 Chaincode code package identifier: fabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2
===== Chaincode is installed on peer1.districtMun.workplace =====
2021-07-06 09:59:43.802 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 001 Installed remotely: response=<status:200 payload:"\nfabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2,022\013fabtender 1" >
2021-07-06 09:59:43.802 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 002 Chaincode code package identifier: fabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2
===== Chaincode is installed on peer0.investigatorFir.workplace =====
2021-07-06 09:59:47.157 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 001 Installed remotely: response=<status:200 payload:"\nfabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2,022\013fabtender 1" >
2021-07-06 09:59:47.157 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 002 Chaincode code package identifier: fabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2
===== Chaincode is installed on peer1.investigatorFir.workplace =====
2021-07-06 09:59:51.678 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 001 Installed remotely: response=<status:200 payload:"\nfabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2,022\013fabtender 1" >
2021-07-06 09:59:51.678 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 002 Chaincode code package identifier: fabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2
===== Chaincode is installed on peer0.auditorFir.workplace =====
2021-07-06 09:59:53.668 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 001 Installed remotely: response=<status:200 payload:"\nfabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2,022\013fabtender 1" >
2021-07-06 09:59:53.668 SAST [cli.lifecycle.chaincode] submitInstallProposal -> INFO 002 Chaincode code package identifier: fabtender 1:a478ea134770298b652f227d0fa500272c06f20ce9dda4d3ce6a22ca00b9f2
===== Chaincode is installed on peer1.auditorFir.workplace =====
garden@gardon-VirtualBox:~/Documents/mscProjects $

```

Figure C.8 Installing chaincode

This section explored the results of the command lines that seek to install the chaincode to all the peer nodes within the Blockchain network. Note that the process of installing the chaincode in each of the peer nodes, allows these peer nodes to generate the chaincode package ID that can be used by the Blockchain network to approve their chaincode. The chaincode package ID determine which chaincode should be used within a specific communication channel. Hence, the following section focuses on the results generated during the process of approving the chaincode installed within the peer nodes.

C.3.2 Approve chaincode

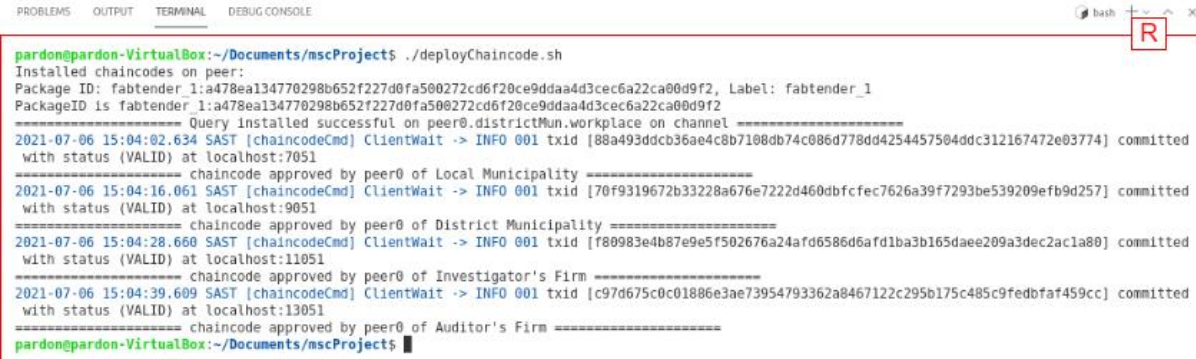
This section focuses on the results generated during the process of approving the chaincode within the Blockchain network. Therefore, lines 128-163 of Figure C.9 depict a function called *approveOrganisations* that contain the command lines used to approve the chaincode within the Blockchain network, while label R represents the results displayed within the command prompt after executing these functions. Line 129 Figure C.9 seeks to call a function called

queryInstalled that contains information related to the chaincode package identifier generated during the installation of the chaincode as discussed in the previous section. This information is used to approve the chaincode that was installed in the peer nodes since it seeks to identify the chaincode used by a specific communication channel. For instance, lines 131-133 of Figure C.9 depict the command line used by the Blockchain network to approve the chaincode installed within peer0 of the Local Municipality. Line 134 seeks to separate the results of this command line from others as displayed within the command prompt after executing the *approveOrganisations* function using line 151. Note that the command line makes use of information such as ordering node, communication channel, and package ID to approve the chaincode installed within a specific peer node. The notation applied to a command-line represented by lines 131-133 can also be applied to the remaining command lines that seek to approve the chaincode installed within other peer nodes and their results are also displayed in label R of Figure C.9.

```

128 approveOrganisations() {
129     queryInstalled
130     setGlobalsForPeer0LocalMunicipality
131     peer lifecycle chaincode approveformyorg -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
132         --channelID $CHANNEL_NAME --name ${CC_NAME} --version ${VERSION} --package-id ${PACKAGE_ID} --sequence ${VERSION} \
133         --tls --cafile $ORDERER_CA --init-required
134     echo "----- chaincode approved by peer0 of Local Municipality -----"
135     setGlobalsForPeer0DistrictMunicipality
136     peer lifecycle chaincode approveformyorg -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
137         --channelID $CHANNEL_NAME --name ${CC_NAME} --version ${VERSION} --package-id ${PACKAGE_ID} --sequence ${VERSION} \
138         --tls --cafile $ORDERER_CA --init-required
139     echo "----- chaincode approved by peer0 of District Municipality -----"
140     setGlobalsForPeer0InvestigatorFirm
141     peer lifecycle chaincode approveformyorg -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
142         --channelID $CHANNEL_NAME --name ${CC_NAME} --version ${VERSION} --package-id ${PACKAGE_ID} --sequence ${VERSION} \
143         --tls --cafile $ORDERER_CA --init-required
144     echo "----- chaincode approved by peer0 of Investigator's Firm -----"
145     setGlobalsForPeer0AuditorFirm
146     peer lifecycle chaincode approveformyorg -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
147         --channelID $CHANNEL_NAME --name ${CC_NAME} --version ${VERSION} --package-id ${PACKAGE_ID} --sequence ${VERSION} \
148         --tls --cafile $ORDERER_CA --init-required
149     echo "----- chaincode approved by peer0 of Auditor's Firm -----"
150 }
151 approveOrganisations

```



```

pardon@pardon-VirtualBox:~/Documents/nscProject$ ./deployChaincode.sh
Installed chaincodes on peer:
Package ID: fabtender_1:a478ea134770298b652f227d0fa500272cd6f20ce9ddaa4d3cec6a22ca00d9f2, Label: fabtender_1
PackageID is fabtender_1:a478ea134770298b652f227d0fa500272cd6f20ce9ddaa4d3cec6a22ca00d9f2
-----
Query installed successful on peer0.districtMun.workplace on channel -----
2021-07-06 15:04:02.634 SAST [chaincodeCmd] ClientWait -> INFO 001 txid [88a493ddcb36ae4c8b7108db74c086d778dd4254457504ddc312167472e03774] committed
with status (VALID) at localhost:7051
-----
chaincode approved by peer0 of Local Municipality -----
2021-07-06 15:04:16.061 SAST [chaincodeCmd] ClientWait -> INFO 001 txid [70f9319672b33228a676e722d460dbcfec7626a39f7293be539209efb9d257] committed
with status (VALID) at localhost:9051
-----
chaincode approved by peer0 of District Municipality -----
2021-07-06 15:04:28.660 SAST [chaincodeCmd] ClientWait -> INFO 001 txid [f80983e4b87e9e5f502676a24afd6586d6ard1ba3b165dae209a3dec2ac1a80] committed
with status (VALID) at localhost:11051
-----
chaincode approved by peer0 of Investigator's Firm -----
2021-07-06 15:04:39.609 SAST [chaincodeCmd] ClientWait -> INFO 001 txid [c97d675c0c01886e3ae73954793362a8467122c295b175c485c9fedbfaf459cc] committed
with status (VALID) at localhost:13051
-----
chaincode approved by peer0 of Auditor's Firm -----
pardon@pardon-VirtualBox:~/Documents/nscProject$

```

Figure C.9 Approve chaincode

This section explored the results of the command lines that seek to approve the chaincode within the ShareTendPro network. Note that the approval of the chaincode installed within the peer nodes allows the Blockchain network to acknowledge the chaincode that was approved

by various organisations or peer nodes by committing the chaincode. Hence, the following section focuses on the results generated during the process of committing the chaincode within the Blockchain network.

C.3.3 Commit chaincode

This section focuses on the results generated during the process of committing the chaincode within the Blockchain network. Additionally, this process seeks to check the organisations that were approved to be part of the communication channel that uses that the chaincode. Therefore, lines 155-175 of Figure C.10 depict a function called *comitChaincodeDefinition* which contains the command lines that allow the Blockchain network to commit the chaincode approved in the previous section, while label R represents the results generated after executing this function (as presented in line 176). However, lines 157–159 depict the command line that seeks to check the readiness of the Blockchain network to commit the chaincode by checking whether all the organisations have been approved or not. Additionally, lines 163-171 of Figure C.10 represent the command line that seeks to commit the chaincode within the Blockchain network, while line 173 represents the command line that seeks to query the committed chaincode as part of establishing whether the chaincode has been committed successfully or not. Lines 166–169 depict the details of the peer nodes that were used to commit the chaincode within the Blockchain network. As indicated in label R of Figure C.10, the results of a check commit readiness command line shows that all the organisations are ready to commit the chaincode. The results of a command line that seeks to commit the chaincode are represented in the middle whereby all the anchor peer nodes have committed the chaincode, while the results of querying the committed chaincode show that all the organisations have endorsed or committed the chaincode.

```

155  commitChaincodeDefination() {
156      setGlobalsForPeer@LocalMunicipality
157      peer lifecycle chaincode checkcommitreadiness \
158          --channelID $CHANNEL_NAME --name ${CC_NAME} --version ${VERSION} \
159          --sequence ${VERSION} --tls --cafile $ORDERER_CA --init-required --output json
160      echo "===== checking commit readiness from Local Municipality ====="
161
162
163      setGlobalsForPeer@AuditorFirm
164      peer lifecycle chaincode commit -o localhost:7050 --ordererTLSHostnameOverride orderer1.orderer.org.workplace \
165          --channelID $CHANNEL_NAME --name ${CC_NAME} --version ${VERSION} --sequence ${VERSION} \
166          --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA --init-required \
167          --peerAddresses localhost:7051 --tlsRootCertFiles $PEER0_LocalMun_CA \
168          --peerAddresses localhost:9051 --tlsRootCertFiles $PEER0_DistrictMun_CA \
169          --peerAddresses localhost:11051 --tlsRootCertFiles $PEER0_InvestigatorFir_CA \
170          --peerAddresses localhost:13051 --tlsRootCertFiles $PEER0_AuditorFir_CA
171      echo "===== chaincode committed by all the anchor peer nodes ====="
172
173      setGlobalsForPeer@DistrictMunicipality
174      peer lifecycle chaincode querycommitted --channelID $CHANNEL_NAME --name ${CC_NAME} --cafile $ORDERER_CA
175  }
176  commitChaincodeDefination

```

```

pardon@pardon-VirtualBox:~/Documents/mscProject$ ./deployChaincode.sh
{
  "approvals": {
    "AuditorFirMSP": true,
    "DistrictMunMSP": true,
    "InvestigatorFirMSP": true,
    "LocalMunMSP": true
  }
}
===== checking commit readiness from Local Municipality =====
2021-07-06 15:09:54.930 SAST [chaincodeCmd] ClientWait -> INFO 001 txid [8a5e7fa6907c45e2529ab57169b77693ea374040a7718b6136823c7819606124] committed
with status (VALID) at localhost:9051
2021-07-06 15:09:55.309 SAST [chaincodeCmd] ClientWait -> INFO 002 txid [8a5e7fa6907c45e2529ab57169b77693ea374040a7718b6136823c7819606124] committed
with status (VALID) at localhost:13051
2021-07-06 15:09:55.791 SAST [chaincodeCmd] ClientWait -> INFO 003 txid [8a5e7fa6907c45e2529ab57169b77693ea374040a7718b6136823c7819606124] committed
with status (VALID) at localhost:7051
2021-07-06 15:09:55.911 SAST [chaincodeCmd] ClientWait -> INFO 004 txid [8a5e7fa6907c45e2529ab57169b77693ea374040a7718b6136823c7819606124] committed
with status (VALID) at localhost:11051
===== chaincode committed by all the anchor peer nodes =====
Committed chaincode definition for chaincode 'fabtender' on channel 'mychannel':
Version: 1, Sequence: 1, Endorsement Plugin: escc, Validation Plugin: vscc, Approvals: [AuditorFirMSP: true, DistrictMunMSP: true, InvestigatorFirMSP: true, LocalMunMSP: true]
===== committing query on Peer@ of District Municipality =====
pardon@pardon-VirtualBox:~/Documents/mscProject$

```

Figure C.10 Commit chaincode

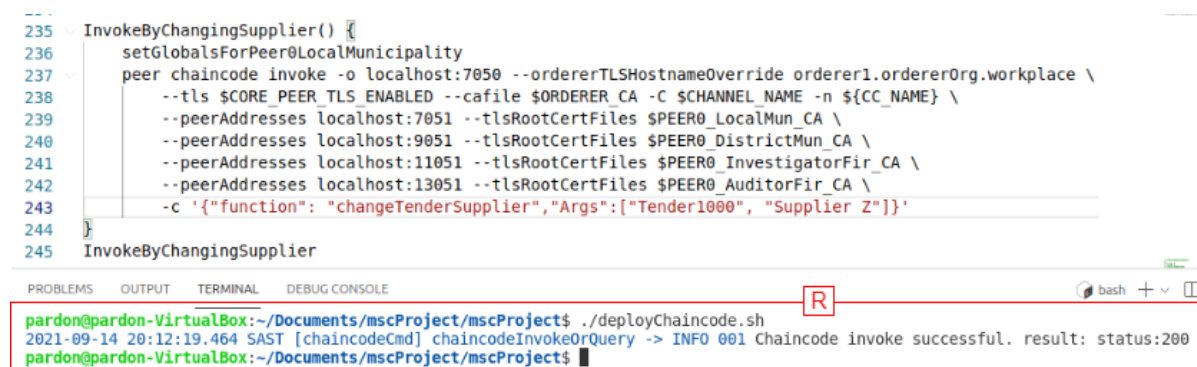
This section detailed the results generated after executing the command lines that seek to commit the chaincode within the Blockchain network. Note that after the Blockchain network has committed the chaincode then the peer nodes would be able to use the chaincode to interact with the Blockchain network or data. Hence, the following section explores the results generated during the process of invoking the chaincode within the Blockchain network.

C.4 Update tender

This section focuses on the results generated by a process that can be used to update the project information stored within the ShareTendPro network. However, this section explores the two possible updating processes implemented within the ShareTendPro network which are “*update tender supplier*” and “*update tender report*”. The first process, which is to *update the tender supplier* focuses on the results generated during a process that seeks to update the details of the supplier (e.g. contact details or to assign a new supplier altogether). The second process, which is the *update tender report* focuses on the results generated by a process that seeks to update the report of the tendering project (e.g., doing editorial updates).

C.4.1 Update Tender Supplier

This section focuses on the results generated by a process that seeks to update supplier information on tendering project X. For instance, lines 235-244 of Figure C.11 depict a function called *InvokeByChangingSupplier()*, which is a function used by peer0 (i.e., computer LM_N0) of the Local Municipality to update the details of a supplier from “Supplier S” to “Supplier Z”. As indicated in lines 237-243, the command makes use of a function called *changeTenderSupplier()*, as seen in line 243, contained within the smart-contract to interact with the Blockchain network. The *changeTenderSupplier()* function requires two arguments. The first argument represents a key “Tender1000” used to identify a tendering project, while the second argument is used to assign the updated details of the supplier which is “Supplier Z” in this case. Therefore, the results displayed (as presented by label R) are generated after executing the *InvokeByChangingSupplier()* function using line 245. The status of the results reflected within label R is 200, which implies that the supplier details were updated successfully.



```

235 InvokeByChangingSupplier() {
236   setGlobalsForPeer0LocalMunicipality
237   peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer1.orderer.org.workplace \
238     --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA -C $CHANNEL_NAME -n ${CC_NAME} \
239     --peerAddresses localhost:7051 --tlsRootCertFiles $PEER0_LocalMun_CA \
240     --peerAddresses localhost:9051 --tlsRootCertFiles $PEER0_DistrictMun_CA \
241     --peerAddresses localhost:11051 --tlsRootCertFiles $PEER0_InvestigatorFir_CA \
242     --peerAddresses localhost:13051 --tlsRootCertFiles $PEER0_AuditorFir_CA \
243     -c '{"function": "changeTenderSupplier", "Args": ["Tender1000", "Supplier Z"]}'
244 }
245 InvokeByChangingSupplier

```

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
2021-09-14 20:12:19.464 SAST [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ █

```

Figure C.11 Change Tender Supplier

After updating the details of the supplier, the second process that seeks to update the details of a project report is considered. Hence, the following section explores the results generated by a process that seeks to update the project report of a tendering project.

C.4.2 Update Tender Report

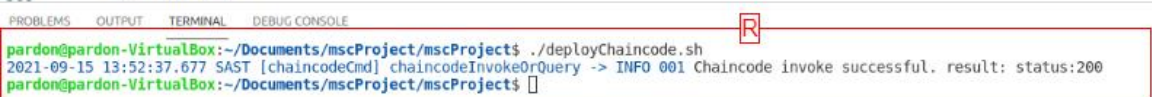
This section explores the details of the results generated by a process that seek to update the details of the project report of tendering project X. Therefore, lines 255-264 of Figure C.12 depict the details of a function called *InvokeByChangingReport()*, which is a function that contains a command line that seeks to update the report of a tendering project associated with the following key “Tender1000”. As indicated in line 263, the command line (represented by

lines 257-263) makes use of a function called *changeTenderReport()* contained within the smart-contract to update a project report of a tendering project associated with the key “Tender1000”, which is project X. Therefore, label R depicts the results generated after executing *InvokeByChangingReport()* using line 265. Note that this function seeks to change the project report to reflect the new report that seeks to portray the following progress “50% of the project was completed within four months”. The status results presented in label R is 200, which implies that the project report was updated successfully.

```

255 InvokeByChangingReport() {
256     setGlobalsForPeer@LocalMunicipality
257     peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer1.ordererOrg.workplace \
258         --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA -C $CHANNEL_NAME -n ${CC_NAME} \
259         --peerAddresses localhost:7051 --tlsRootCertFiles $PEER0_LocalMun_CA \
260         --peerAddresses localhost:9051 --tlsRootCertFiles $PEER0_DistrictMun_CA \
261         --peerAddresses localhost:11051 --tlsRootCertFiles $PEER0_InvestigatorFir_CA \
262         --peerAddresses localhost:13051 --tlsRootCertFiles $PEER0_AuditorFir_CA \
263         -c '{"function": "changeTenderReport", "Args": ["Tender1000", "50% of the project was completed within four months"]}'
264 }
265 InvokeByChangingReport

```



```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
2021-09-15 13:52:37.677 SAST [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$

```

Figure C.12 Change Tender Report

C.5 Delete tender

This section focuses on the results generated by a process that seeks to delete project information from the ShareTendPro network. Therefore, lines 265-271 of Figure C.13 depict the details of a function called *InvokeByDeletingTender()*, which contained a command line that seeks to delete project information of a tendering project associated with the key “Tender1000”. As indicated in line 271, the command line (represented by lines 265-271) makes use of a function called *deleteTenders()* contained within the smart-contract to interact with the Blockchain network. Note that the *deleteTenders()* requires one argument which is the key “Tender1000” of a particular tendering project, which is project X in this case. Label R of Figure C.13 represents the result generated after executing *InvokeByDeletingTender()* function using line 273. As indicated in label R, the command line returns a status of 200 which implies that the project information was successfully deleted within the world-state of the ledger. This implies that the project information will no longer be accessible using a normal *query process*, however, it can only be accessible using a process that seeks to *query the project history* of that tendering project.

```

263 InvokeByDeletingTender() {
264     setGlobalsForPeer@LocalMunicipality
265     peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer1.orderer1.org.workplace \
266     --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA -C $CHANNEL_NAME -n ${CC_NAME} \
267     --peerAddresses localhost:7051 --tlsRootCertFiles $PEER0_LocalMun_CA \
268     --peerAddresses localhost:9051 --tlsRootCertFiles $PEER0_DistrictMun_CA \
269     --peerAddresses localhost:11051 --tlsRootCertFiles $PEER0_InvestigatorFir_CA \
270     --peerAddresses localhost:13051 --tlsRootCertFiles $PEER0_AuditorFir_CA \
271     -c '{"function": "deleteTenders", "Args": ["Tender1000"]}'
272 }
273 InvokeByDeletingTender

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
2021-09-14 20:19:22.175 SAST [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ █

```

Figure C.13 Delete Tender

After performing all the processes that seek to either create, update or delete the project information of a particular tendering project, then the evidence related to such processes is stored within the transaction logs. The information stored within the transaction logs is immutable since it seeks to preserve the evidence of what transpired within that specific tendering project. Additionally, the evidence contained within the transactional logs can also be used as forensic data since it portrays the entire history of a particular tendering project. Therefore, the following section focuses on the results generated by a process that seeks to access a project history of tendering project X.

D. Derived Publications

The following item contains the list of the publications and poster derived from this dissertation.

1. P.T. Ramazhamba, H.S. Venter. A distributed model for sharing tendering problem information in the South African Local Government. In the CSIR Emerging Researchers Symposium (CSIR-ERS) Poster. 2022
2. P.T. Ramazhamba, H.S. Venter. A distributed model for sharing tendering problem information in the South African Local Government. In the 21st International Conference on Cyberworlds (CW2022), 2022.
3. P.T. Ramazhamba, H.S. Venter. Using distributed ledger technology for digital forensic investigation purposes on tendering projects. International Journal of Information Technology (Journal paper was accepted).
4. P.T. Ramazhamba, H.S. Venter. ShareCrime: A Blockchain concept for sharing criminal information in the South African Criminal Justice System. (Accepted by IFIP WG 11.9 International Conference on Digital Forensics).

The following sections contain the detail of these publications.

D.1. A distributed model for sharing tendering problem information in the South African Local Government. In the CSIR Emerging Researchers Symposium (CSIR-ERS) Poster

Abstract

The South African Local Government (SALG) uses the tendering system to promote social and industrial/environmental policies. Some of the project information shared by these suppliers during the tender bidding process plays a critical role when it comes to awarding a tender project to a particular supplier since it reflects their competency area and project history. Some of the tools used to share this project information are reports, meetings, presentations and site visits. Therefore, this study proposes a distributed model that might be used to share tendering project information securely and efficiently with all the parties that have an interest in it. The proposed model seeks to promote the need for sharing project information, while eliminating issues that are related to a single point of failure or having an organisation that has central powers over project information. Additionally, the proposed model can also be used to foster collaboration between the public and private sectors by becoming an essential tool that might be used to securely share project information without colluding.

Introduction

Figure 1 depicts the current tendering system concept (CTSC) used by the District and Local Municipalities to share project information with all the parties that have an interest in it. The CTSC is structured in a centralised manner, whereby the municipalities are seen as the centre that distributes project information, as shown in Figure 1. Note that project information plays an important role when it comes to awarding a tender to a particular supplier since it reflects the competency area and project history of a particular supplier. All the suppliers are required to submit such information when they are applying for a tender project [1]. Some of this information will go through a verification process. Some of the tools used to share project information are reports, meetings, presentations and site visits, as well as trusted third parties. Through these processes, project information might be altered for corrupt purposes at any given stage. Therefore, the primary problem of this study is that paperwork is used to share project information, which might contribute to the illicit altering of information during the process. This might also affect the fairness, transparency, data integrity and competitiveness of the tendering system.

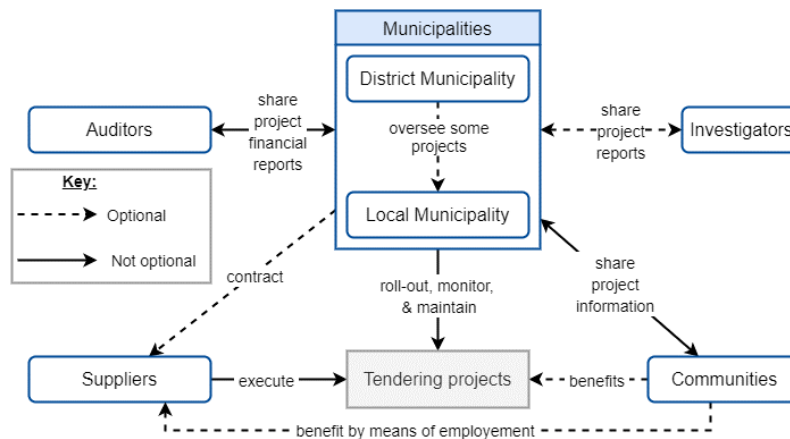


FIGURE 1: CURRENT TENDERING SYSTEM CONCEPT

Technology description

Blockchain technology (BCT) is regarded as a technology solution used to implemented distributed and shared ledger systems for storing a wide range of assets or transactions [2]. However, the proposed solution seeks to share project data among various parties that are known; therefore, a private Blockchain that does not use any cryptocurrency or mining algorithms to add new transactions will be suitable for this study. Hence, HLF is a favorable BCT framework adopted by this study since it was designed to develop a new generation of transactional applications that are aimed at establishing trust, accountability, and transparency [3,4]. HLF is an open-source private Blockchain framework that can be used to implement cross-industrial Blockchain solutions [5].

Share tendering project (ShareTendPro) model

The proposed model aimed at securely and efficiently distributing project information among all the parties that have an interest in it. Figure 2 depicts the components of the ShareTendPro model interact with each other.

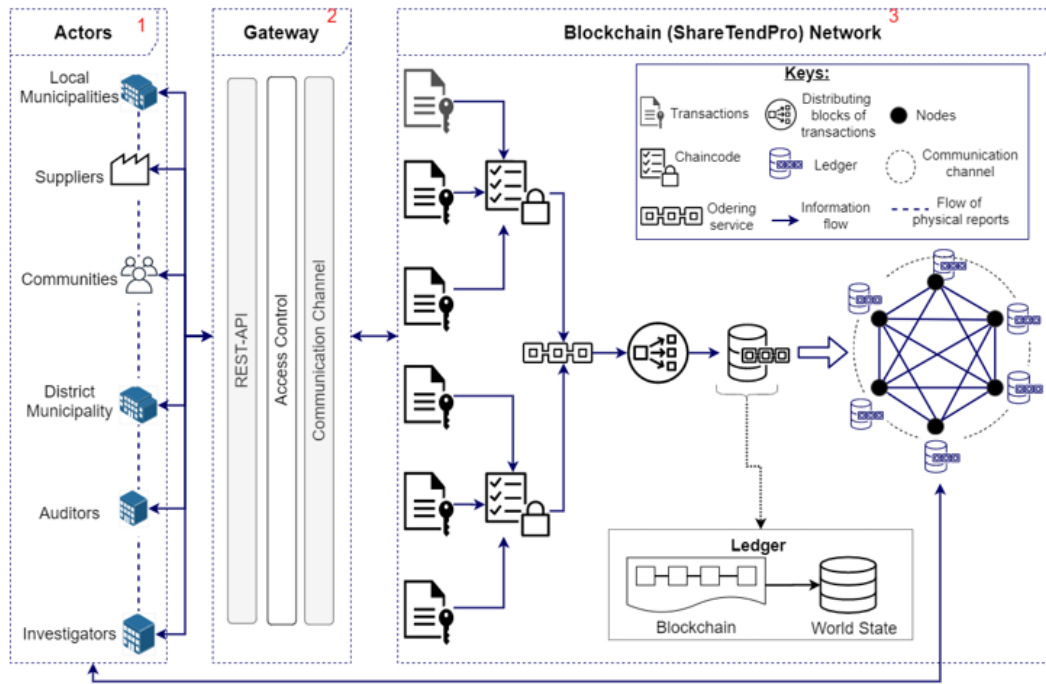


FIGURE 2: SHARETENDPRO MODEL

```

296 chaincodeQueryProjectHistory()
297     echo "===== query chaincode using Peer0 Auditor's Firm ====="
298     setGlobalsForPeer0DistrictMunicipality
299     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} \
300     -c '{"function": "getHistoryForTender", "Args": ["Tender1000"]}'
301     echo "=====end query===== "
302 }
303 chaincodeQueryProjectHistory
    
```



```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
===== query chaincode using Peer0 Auditor's Firm =====
[{"TxId": "a35fa2ec5f2225a5befcdb3936facb3cac0c194d3d3ac215a67c2f251616be68", "Value": null, "Timestamp": "2021-09-15 12:28:09.409964752 +0000 UTC", "IsDelete": "true"}, {"TxId": "919896445f5350fbeb11c3011140090b5219d9593b9fd7643dac91ed6d31030f", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "50% of the project was completed within four months", "supplier": "Supplier Z"}, "Timestamp": "2021-09-15 12:24:46.098319371 +0000 UTC", "IsDelete": "false"}, {"TxId": "b31185f27ae9cd52119e1e81d7fdbdc2b8e369281e15f227748c6781681e9e9c", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier Z"}, "Timestamp": "2021-09-15 12:28:23.951643768 +0000 UTC", "IsDelete": "false"}, {"TxId": "94a208cc045ee21011580be40f2241e528381eeb59f78bc505a1a75517da258c", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier S"}, "Timestamp": "2021-09-15 12:21:58.628973647 +0000 UTC", "IsDelete": "false"}]
=====end query=====
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$
    
```

Transaction ID for deleteTender

Transaction ID for updateTenderSupplier

Transaction ID for updateTenderReport

Transaction ID for createTender

FIGURE 3: SHARETENDPRO RELUSTS

Theoretical use-case scenario

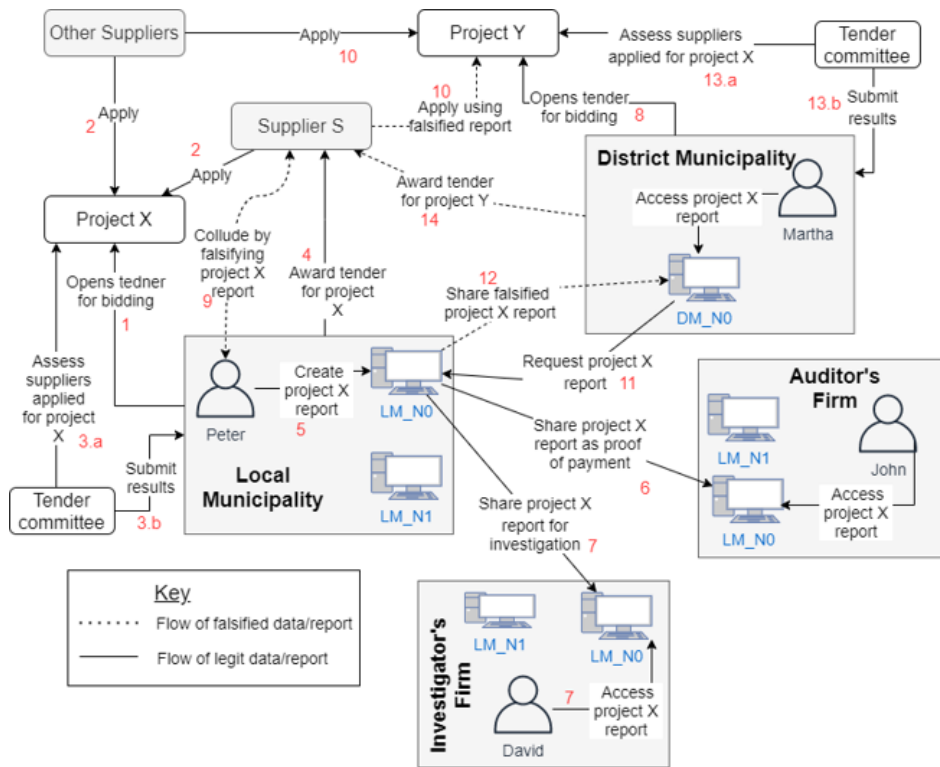


FIGURE 4: FICTIONAL USE-CASE SCENARIO

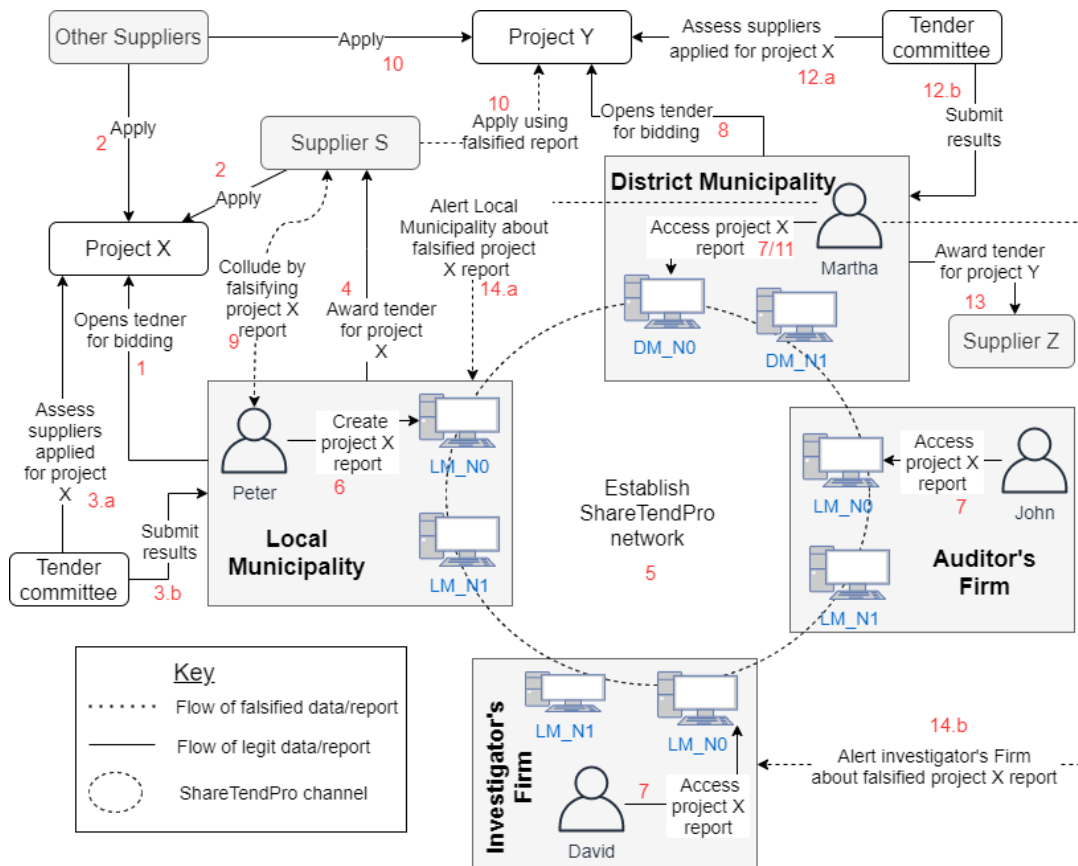


FIGURE 5: SHARETENDPRO SOLUTION

Benefits of the ShareTendPro model

They are various benefits associated with the adoption of the proposed model, and these benefits includes; distributed nature of the ShareTendPro model, enhanced information security, greater transparency over data, time efficiency due to availability of data to all the participants, credible evidence, promotes real-time auditing and investigations, and promote collaboration.

Conclusion

The SALG uses a tendering system to promote public and private partnership; therefore, the proposed model becomes an essential platform for securely sharing project information without colluding. Additionally, the ShareTendPro network ensures that all the participants have instant access to real-time data stored within the Blockchain network without having to worry about issues that emanate from requesting those data directly from a specific organisation because some of them might be reluctant to share them. The ShareTendPro network, therefore, would provide a revolutionary step towards curbing corruption in countries like South Africa, where corruption currently enjoys high tide.

References

1. S. Ngobeni, "An analysis of the tender process in national government in South Africa," in MBA Thesis, North-West University, Potchefstroom, 2011
2. ASTRI, "Whitepaper on Distributed Ledger Technology," 11 November 2016. [Online]. Available: <https://www.astri.org/tdprojects/whitepaper-on-distributed-ledger-technology/>. [Accessed 18 02 2019].
3. Hyperledger, "About," Hyperledger Projects, [Online]. Available: <https://www.hyperledger.org/about>. [Accessed 19 03 2019]
4. C. Cachin, "Architecture of the hyperledger blockchain fabric," in In Workshop on distributed cryptocurrencies and consensus ledgers, 2016.
5. Hyperledger-fabric, "Key concepts: Ledger," Hyperledger-fabric documents release-1.4, [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-1.4/ledger/ledger.html>. [Accessed 09 October 2019]

D.2. A distributed model for sharing tendering problem information in the South African Local Government. In the 21st International Conference on Cyberworlds (CW2022)

Abstract

The South African Local Government uses the tendering system to deliver some of the basic services to the surrounding communities to promote social and industrial or environmental policies. However, this process still relies heavily on a manual process, which requires skilled personnel to deal with the forms and administrations of the entire tendering process. Some of the project information shared by the supplier during the tender bidding process plays a critical role when it comes to awarding a tender project to a particular supplier since it reflects the competency area and project history. Some of the tools used to share this project information are reports, meetings, presentations, and site visits. Therefore, this study proposes a distributed model that might be used to share tendering project information securely and efficiently with all the parties that have an interest in the tendering project. The proposed model seeks to promote the need for sharing project information while eliminating issues that are related to a single point of failure or having an organisation that has central powers over project information. Additionally, the proposed model can also be used to foster collaboration between the public and private sectors by becoming an essential tool that might be used to securely share project information without colluding. The proposed model also incorporates the benefits and promises that come with the adoption of distributed ledger technology (blockchain) as a technology solution.

Keywords—Tendering system, South African Local Government, Project information sharing, Distributed ledger, Blockchain

I. Introduction

The South African organs of state have adopted the use of information and communication technologies (ICTs) as a tool that enables them to perform certain tasks. One of the main reasons for these organs of state to adopt the use of ICTs is that some of their tasks require innovations when it comes to issues related to collecting, processing, and analysing digital information. Digital information can be viewed as data that requires electronic devices such as personal computers or laptops to process and manipulate it. However, some of the tasks of these organs of states still rely heavily on a manual process, whereby they still require the use of paperwork

to achieve certain tasks. For instance, all the South African organs of the state still require their suppliers to submit documents whenever there is a tender bidding process. Some of the information used within this process is regarded as essential because it can be used as a deciding factor when it comes to awarding a tender project. Tendering can be viewed as an essential procedure for some of the organisational operations as some of these organs of the state rely heavily on this process to procure goods and services. Tendering can also be viewed as the central method used by the organs of state to deliver some of the basic services to the surrounding communities with an aim of promoting social and industrial or environmental policies [1]. However, tendering can only be regarded as an essential tool if the procedures and principles that underpin it are adhered to [2].

The process of sharing tendering project information might raise some security concerns because illegal information might be used to influence the decision of the tendering committee who are tasked to award the tendering project. The tendering committee only relies on the documents submitted by the supplier who seeks to participate in the tender bidding process. Therefore, the use of a referee and other methods such as sending reports, meetings, or presentations using electronic mail as a mechanism to confirm whether the information shared with the tender committee is true or not, might raise data integrity concerns, because the information might be altered for corrupt purposes at any given stage. Hence, the current tendering system (CTS) still relies heavily on manual processes, which require skilled personnel to handle the forms and administer the entire process [3]. The primary problem of this study is the use of conventional methods such as meetings, reports, presentations, site visits, etcetera to share project information with the parties that have an interest in the tendering project since these methods are prone to fraudulent actions.

The remainder of this study is structured as follows: *Section II* provides a brief overview of the background concepts related to the tendering system used by the South African Local Government (SALG). *Section III* provides the details that seek to explore the technology description adopted by this study. *Section IV* presents the proposed model used to share tendering project information securely and efficiently. Thereafter, the theoretical use-case of the proposed model is presented in *Section V*. *Section VI* explores details of the related work, which includes comparing them with the proposed model. Finally, the last section, which is *Section VII* details the conclusion, as well as the future work related to this study.

II. Background

The South African Government is comprised of three spheres, namely National, Provincial, and Local Government. The National Government is responsible for overseeing the Provincial Government, while the Provincial Government oversees the Local Government. However, the delineation of this study lies in sharing tendering project information within the Local Government because it is regarded as the smallest sphere used by the South African Government to deliver some of the basic services to their surrounding communities. Some of these basic services or projects have a direct impact on these communities since they might be intended to either develop the surrounding communities or improve the socio-economic standing of that community. The SALG is divided into three types of municipalities, namely Metropolitan (also known as Metro), District, and Local Municipalities as shown in Figure 1. The District and Local municipalities share their responsibilities when it comes to executing some of the projects, while Metros are regarded as standalone municipalities since they report directly to the Provincial government. However, these municipalities use Supply Chain Management (SCM) as a tool that guides the execution of their projects, and the South African National Treasury is responsible for implementing the SCM [4]. The SCM requires these municipalities to have their role players in-place who are responsible for executing these projects to address the issue of accountability. Additionally, all these processes are bounded by the legislative frameworks and pillars of procurement [5]. Therefore, Figure 1 summarises this section by visualising the concepts that interact with tendering projects.

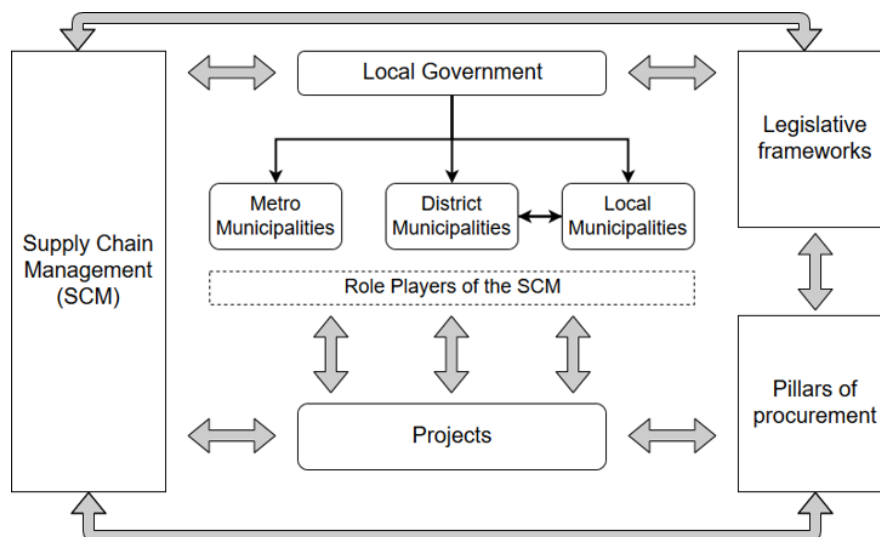


FIGURE 1: LOCAL GOVERNMENT TENDERING PROJECT CONCEPT

The following items seek to provide a high-level overview of the concepts highlighted within Figure 1.

- *SCM*: this is the process that seeks to manage all the activities associated with the procurement process [6]. Note that the procurement process requires suppliers to share some of their information in relation to their competency area and project history when they bid for tendering projects. Additionally, the information shared by the supplier is also used for decision-making purposes, especially when it comes to awarding a specific tender project to a particular supplier.
- *Legislative frameworks*: these are the legislations that seek to govern the procurement processes used by the SALG. The legislation that seeks to govern the procurement processes are *Constitution* [7], *Preferential Procurement Policy Framework Act* [8], and *Municipal Finance Management Act of 2003* [9].
- *Pillars of procurement*: all the procurement legislation are incorporated with the core pillars of procurement to ensure that all the procurement processes are adhered to. The South African Government, through the Public Finance Management Act of 1999 has identified five pillars that need to be considered during the procurement process. These pillars are “*value of money*”, “*open & effective competition*”, “*ethics and fair dealings*”, “*accounting & reporting*”, and “*equity*” [4][10].
- *Role players of the SCM*: these are the individuals accountable for the procurement processes executed by their municipality and these role players are the Municipal Council, Account Officer, and the Municipal SCM Unit [9][11].

The section introduced the concepts associated with the tendering project within the SALG. The following section explores an overview of the CTS used by the SALG with the aim of visualising how various stakeholders interact with the project information of their interest.

A. SALG current tendering system overview

The CTS used by the SALG requires all the municipalities to share some of their project information with the affected parties, such as communities and investigators. Communities act as the beneficiaries of some of these projects, while the investigators are responsible for investigating irregularities that might occur during the execution of some of these projects. Additionally, these municipalities are also required to share their financial reports of these projects with their auditors because they are responsible for overseeing how these municipalities use public funds. The communication channel used by these municipalities to share project information is structured in a centralised manner whereby municipalities are seen as the centre that distributes project information to all the parties that have an interest in the project

information as shown in Figure 2. Furthermore, this communication channel relies heavily on paperwork to share project information, even though some of this information is used for decision-making purposes, especially when it comes to awarding a tender to a particular supplier. Figure 2 seeks to summarise the concept discussed within this section by visualising how the current project information-sharing concept works.

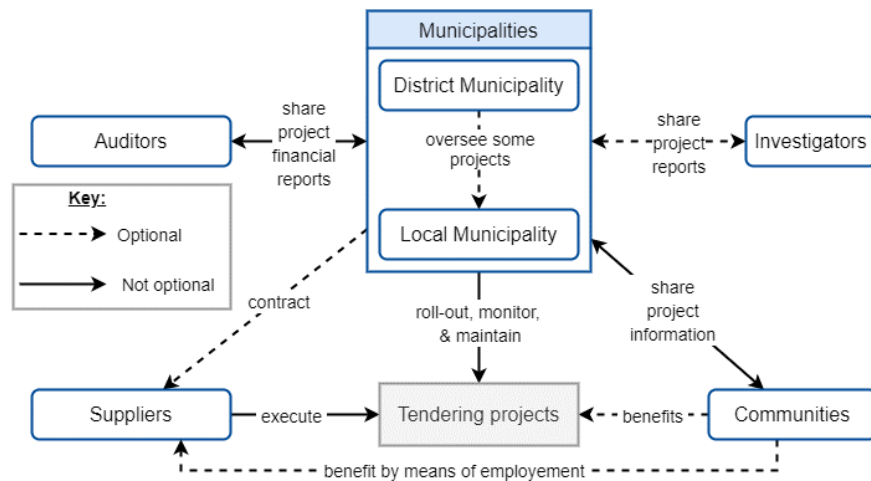


FIGURE 2: CURRENT PROJECT INFORMATION-SHARING CONCEPT

B. The importance of monitoring tendering projects

Tender projects play an important role in stimulating the development of many countries since some of these projects are designed to improve their infrastructure while also empowering the surrounding communities. As indicated in the previous section, the government uses tender projects to deliver some of its services and it also invests a huge amount of money to fund such projects. Therefore, having the legislations, pillars, and role players in place to govern the procurement processes does not ensure the successful implementation of these projects. However, there are some aspects that need to be considered that contribute to getting the best benefits and value for money out of these projects. These aspects are monitoring and assessment of projects. Otieno [12] distinguishes these aspects as follows: Monitoring of projects is “the process that provides the necessary information and ensures the use of such information by management to assess the effects or impact of the projects”. Assessment of projects is drawn from “the use of data generated by the monitoring systems to analyse the impact of the project trends” [12].

These definitions emphasise that the assessment of projects depends on the monitoring tool since it aimed on ensuring whether the desired objectives have been achieved or not. Therefore, this section focuses on the monitoring of projects because this study aimed at sharing project information which falls under the provision of the necessary information for decision-making

purposes. Monitoring of projects can also be viewed as a project management tool that focuses on providing continuous feedback on the project implementations. Some of the reasons behind using this aspect as a project management tool include: the assessment of the project understanding by stakeholders, minimising the risk of project failure, promoting project management, and assessing the progress of the project implementation [12]. The commonly used tools for monitoring tendering projects are verbal communication, meetings, reports, and diary notes. However, all these tools have their own limitations, and they are also vulnerable to data integrity, transparency, and accountability. Project monitoring tools act as mechanisms that lubricate the progress of the project with an aim of achieving the desired objectives [12]. Therefore, it is important to adopt an appropriate monitoring tool that will provide the maximum benefits out of the tendering project.

This section has provided the background details of the concepts that are related to the tendering system as part of trying to examine the tendering system used by the SALG. Additionally, the section provided the importance of monitoring these tendering projects to achieve the desired objectives. Therefore, the following section focuses on the research method adopted in this study.

III. Technology description

There are various technologies that can be adopted to achieve the desired objective of this study. These technologies can be classified based on how they use their ledger systems to either store or share information. These ledger systems have evolved significantly over the past years from a centralised system to where it has now become distributed. Figure 3 depicts the classifications of the three main evolution stages of the ledger systems, which are centralised, decentralised, and distributed. However, this study adopts the distributed ledger system (DLS) because it does not have issues related to a single point of failure. Additionally, it also seeks to share project information among all the parties that have an interest in the project or its information.

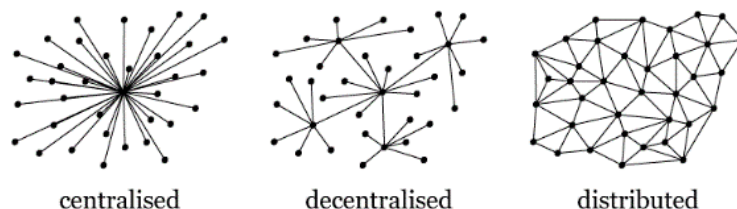


Figure 3: Evolution of ledger systems [13]

ASTRI [14] defined distributed ledger technology (DLT) as a “technology protocol that can be used for developing a replicated and shared ledger system that stores a wide range of assets

and transactions in a distributed manner”. This implies that a DLS is regarded as a shared ledger system since its records of transactions are maintained across several locations or among multiple nodes, regardless of their geographical location [15]. Basically, this means that all the nodes that are found within that network have the same copy of the ledger. Hence, a DLS does not consist of a central repository or a single point of failure like a centralised ledger system. However, every time when a specific node in a DLS has made some valid changes on the ledger, those changes are propagated automatically and shared with other nodes that form part of the network. Additionally, this mechanism of sharing information is also aimed at maintaining data integrity across all the nodes within that network.

Note that the DLT has become more prevalent in 2008, after the circulation of a white paper titled “Bitcoin: A Peer-to-Peer Electronic Cash System” authored by Satoshi Nakamoto [16]. The white paper proposed a solution for the financial industry that addresses the issue of double-spending and eliminating the norm of using intermediaries. However, the ideology of the proposed solution existed theoretically [17, 18], until 2009 when the first DLT implementation (Bitcoin system) emerged by Satoshi Nakamoto [16]. The underlying technology used by Satoshi Nakamoto to implement the Bitcoin system was termed “Blockchain” technology. Blockchain refers to the ways in which the proposed system stores and organises its information. The word “Blockchain” is a combination of two words namely “block” and “chain”. Therefore, DLTs use blocks to store their information, and these blocks are linked together to form a chain-like data structure, hence “Blockchain”. As time progresses, similar ways of organising and storing information emerged which led to the term DLTs as a broad term used to categorise such technologies [15].

IV. A proposed distributed model for sharing tender project information

The proposed model aimed at sharing project information securely and efficiently among various parties that have an interest in the tendering project. Therefore, to achieve this objective, the proposed model must incorporate the following components namely: *actors*, *gateway*, and *Blockchain network*. These components are explored in detail in later sections, however, for the convenience of the reader to understand the basics of the logic behind the proposed mode, the components are briefly explained below:

- *Actors*: are the role players of the proposed model and may for example consist of various organisations and their members.
- *Gateway*: allows actors to interact with the Blockchain network of the proposed model, including the policing mechanism.

- *Blockchain network* stores and distributes project information among all the actors that have an interest in the tendering project.

Figure 4 depicts an overview of how these components (*actors*, *gateway*, and *Blockchain network*) interact with each other.

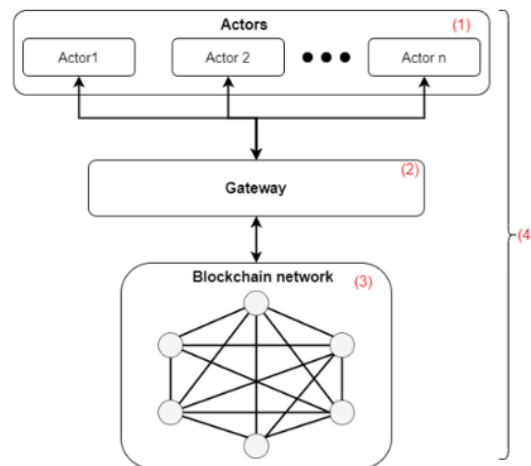


FIGURE 4: SHARETENDPRO MODEL OVERVIEW

As highlighted in Figure 4, this study adopts the following approach to explore how these components work in the proposed model.

1. Identify the actors of the proposed model.
2. Establishing the gateway that will be used to identify and authorise actors as they interact with project information.
3. Establishing the Blockchain network that can be used by the proposed model to securely store and share project information.
4. Defining the ShareTendPro model, which is the integration of steps 1–3 above.

All these steps are discussed in detail in the following subsections to outline the details of this approach.

A. Identify the actors of the proposed model

They are number of actors that might have an interest in the tendering project information and these actors can be classified into two categories namely: *main actors* and *additional actors*. The following items explore the details of these two categories.

1. Main actors: are all the actors that have a direct interaction with the tendering project information and these actors includes:
 - a) *District Municipalities (DMs)*: they are responsible for rolling out, monitoring, and maintaining tender projects that fall under their mandate, including overseeing some of

the projects executed by their Local Municipalities (LMs). Hence, their role within the proposed model will be creating a tendering project, sharing project information, and accessing project information of other municipalities.

- b) *Local Municipalities*: their roles are almost the same as the roles explored within the DM, besides the role of overseeing other projects. Hence, their role within the proposed model is also the same as the roles assigned to the DMs.
- c) *Communities*: they are the beneficiaries and stakeholders of some of these projects. Hence, the municipalities are required to share some of their project information with these communities at some point. Therefore, their role within the proposed model is to access and share project information.
- d) *Suppliers*: these are organisations that seek to render certain services on behalf of these municipalities (i.e., DMs or LMs). Hence, all these Suppliers report directly to the municipality which awarded them the tender. Therefore, their role within the proposed solution is to create or share project information.

2. Additional actors: are all the actors that have an indirect interaction with the tendering project information and these actors are:

- a) *Auditors*: are responsible for ensuring that municipalities account for their actions by auditing their financial expenditures to check for irregularities and misuse of public funds. Hence, their role within the proposed model is to access project reports and share their audit reports.
- b) *Investigators*: are responsible for gathering all the possible evidence that identifies the occurrence of illegal activities within a tendering project. Hence, their role within the proposed model is to access project information related to their investigations.

Figure 5 depicts the interaction of the following actors with the tendering project information: DM, LMs, Auditors, Investigators, Communities, and Suppliers.

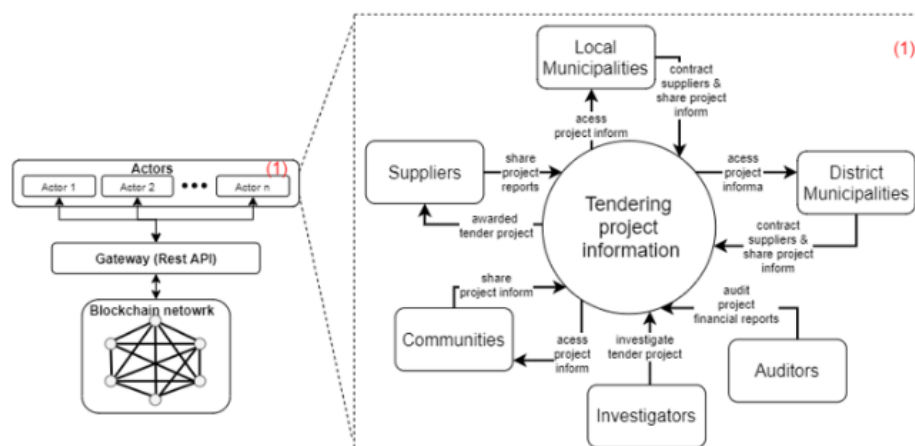


FIGURE 5: ACTOR COMPONENT

The following section explores the gateway component as used by these actors to interact with the tendering project information stored within the proposed model.

B. Establishing the gateway

The Blockchain technology uses a gateway component to separate the role played by various actors within the network. However, some of the Blockchain frameworks achieve this by using the following mechanisms:

- *REST-API*: allows various actors to use an application programming interface (API) to interact with the Blockchain network. In other words, this process exposes the deployed network as a REST-API that allows authenticated actors to interact with the Blockchain data using queries. All the transactions submitted through the REST-API are assigned an HTTP request operation which either creates, reads, updates, or deletes data stored within the network. In addition, all these transactions will also be assigned a digital certificate to preserve non-repudiation.
- *Access control list (ACL)*: manages the access rights of all the authorised actors as they interact with the Blockchain data. These access rights can be categorised into two namely read and write access rights. For instance, communities, suppliers, auditors, and investigators are not allowed to write or create tendering project information, however, they are allowed to read or view some of the details contained within it.
- *Secure communication channels*: allow a specific group of actors to secretly share project information. For instance, a channel might be created for certain LMs that fall under a specific District to share project information since some of their tendering projects are overseen by a particular DM.

Figure 6 represents the above mechanisms, i.e., *REST-API*, *ACL*, and *secure communication channels*, used by the proposed model to manage the identities of various actors, including providing access to the Blockchain network.

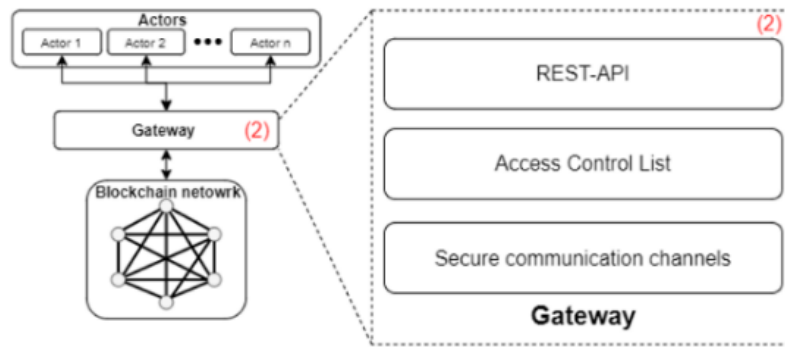


FIGURE 6: GATEWAY COMPONENT

The following section explores how the Blockchain network component works, including how project information is distributed among various actors or nodes within the network.

C. ShareTendPro model as a Blockchain network

This component focuses on the operational concept or logic behind storing and sharing project information with all the actors that have an interest in the tendering project. This component achieves this by allowing all the authorised actors to submit project information as transactions. However, all these transactions should meet specific requirements associated with it. Hence, the Blockchain network makes use of the smart contract (SC) to govern all the transactions within the network. The SC consists of predefined conditions associated with each transaction and all the transactions that do not meet such requirements are discarded or declared as rejected by the network.

All the accepted transactions are forwarded to the ordering service for ordering. The ordering service collects all the accepted transactions and groups them into blocks, which are then distributed among all the nodes within the network. The Blockchain network component achieves this by using a DLS that allows it to distribute these blocks of transactions to various nodes. However, each node will then make use of the SC to verify these ordered transactions before appending them to the ledger. Once this process is complete and all the nodes have appended the new transactions to their ledger, then all the actors who have an interest in that tendering project will now have access to the updated project information. Figure 7 depicts how the Blockchain network component distributes project information among various nodes or actors. Part A of Figure 7 represents the information flow, while part B represents the distributed nature of the nodes or actors as they share project information.

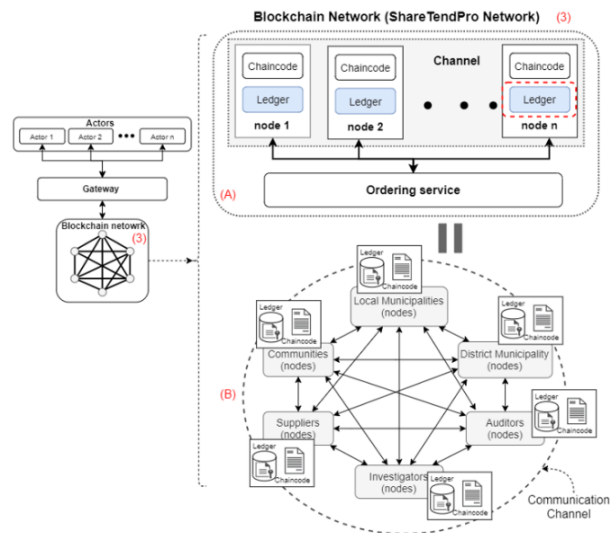


FIGURE 7: BLOCKCHAIN NETWORK COMPONENT

The following section seeks to integrate all these components (*actors*, *gateway*, and *Blockchain network*) to generate the final step labelled number 4 as shown in Figure 4.

D. ShareTendPro model as an integrated whole

This section integrates the components discussed in Figure 4 to generate a ShareTendPro model as our last step. Therefore, Figure 8 depicts a graphical representation of the ShareTendPro model as an integral of these components (*actors*, *gateway*, and *blockchain network*). It also reflects the flow of the project information as it passes through various components and objects. The numbers labelled 1-3 represent the three respective components, while number 4 can be viewed as the approach used by this study to explore how the proposed model integrates. The ShareTendPro model allows various actors to share tendering project information securely and efficiently. The Blockchain network component is one of the main key components that allow the ShareTendPro model to achieve its objectives because it is responsible for storing and sharing project information securely and efficiently.

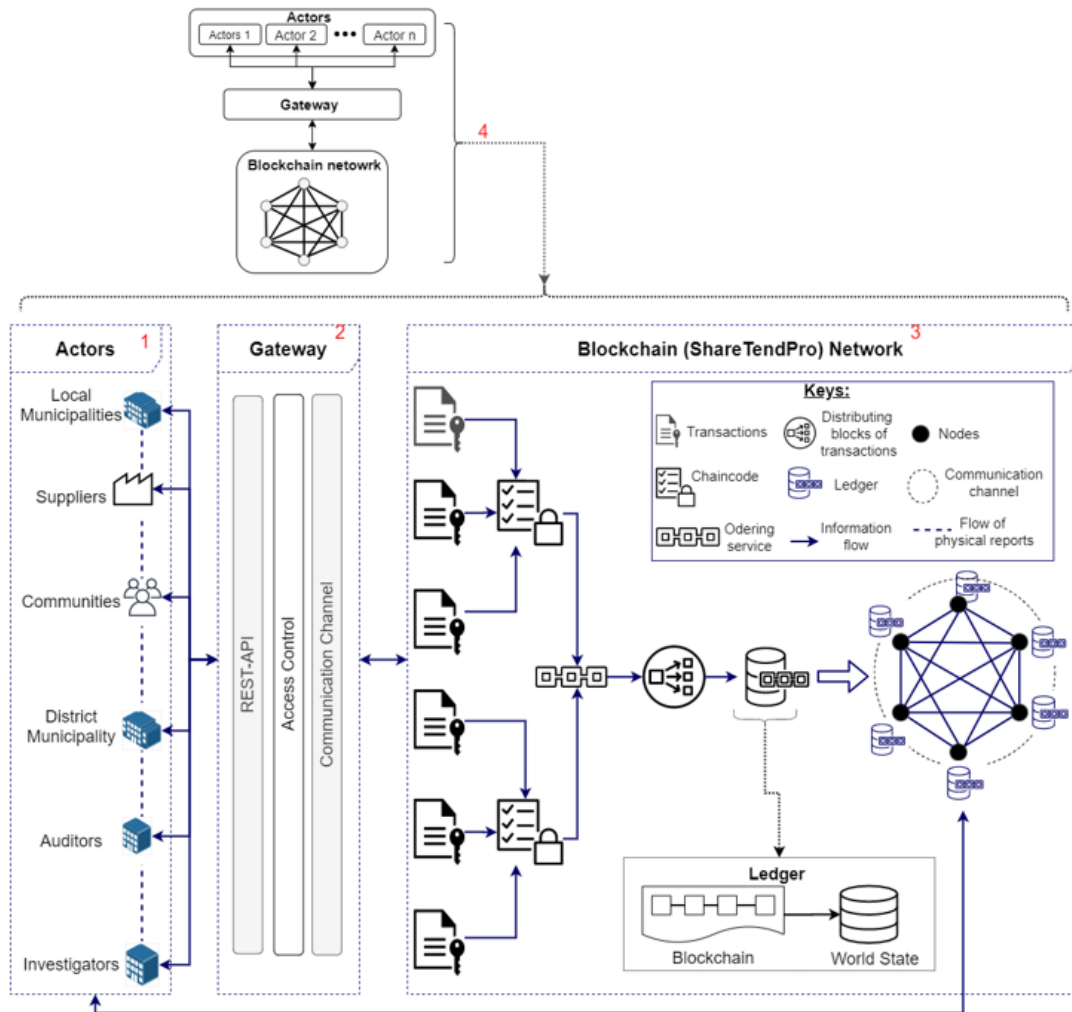


FIGURE 8: SHARETENDPRO MODEL

V. Theoretical use case scenario

This section explores the scenario that could be addressed by the proposed solution. Therefore, the following items present the process that takes place within the scenario as shown in Figure 9.

1. Step 1: the LM opens tendering project X for bidding.
2. Step 2: various suppliers apply for tender project X by submitting tender documents to the LM.
3. Step 3: the tendering committee assigned by the LM assesses all the suppliers who applied for project X and submits the results of the assessment to the LM.
4. Step 4: the LM awards project X to supplier S based on the outcomes presented by the tendering committee.
5. Step 5: the LM assigns Peter to manage project X. Thereafter, Peter uses computer LM_N0 (which stands for Local Municipality node 0) to issue a progress report for

project X as part of his responsibilities which seeks to portray the following progress “so far, 20% of project X was completed within four months”.

6. Step 6: Peter shared this report with John from the Auditor’s Firm who was tasked to audit the financial expenditure of tendering project X. Hence, the report acts as proof of payments associated with the work that was completed by supplier S.
7. Step 7: Peter also shared this report with David from the Investigator’s Firm (IF) who was tasked to investigate allegations of corruption in the tendering project X. The report acts as proof of work completed by supplier S.
8. Step 8: later on, the DM opens tender project Y for bidding. Assume that project Y is similar to project X.
9. Step 9: assume that supplier S decided to collude with Peter when it comes to falsifying the report of project X to portray the following progress “50% of project X was completed within four months”.
10. Step 10: various suppliers apply for project Y, including supplier S. Assume that supplier S has included a falsified progress report of project X when applying or bidding for project Y and included Peter as a referee who can provide more clarifications regarding project X.
11. Step 11: the DM assigns Martha from the tendering committee of project Y a task to request a progress report of project X from Peter as part of trying to confirm whether Supplier S managed to complete 50% of the project within four months or not. Note that Martha used computer DM_N0 (which stands for District Municipality node 0) to send an electronic mail (email) to Peter when requesting the progress report of project X.
12. Step 12: Peter submitted a falsified progress report of project X to Martha (DM_N0) at the DM.
13. Step 13: The tendering committee of the DM assesses all the suppliers who applied for project Y and submits the results of the assessment to the DM.
14. Step 14: the DM awards tendering project Y to Supplier S based on the outcome of the assessment which was motivated by the information provided by the supplier and confirmed by Peter who works at the LM.

The main objective of this scenario was to depict a loophole that might be used to tamper with the project information in such a way that it can be used to influence the decision of other projects offered by a different municipality. For instance, in the scenario, a falsified report of project X was used to influence the decision when it comes to awarding project Y offered by

the DM. Figure 9 seeks to visualise this scenario as various people in different organisations interact with either a falsified or a legit report of project X. Assume that the communication mechanism used to share the report of project X was an email. Hence, Figure 9 depicted the computers used by various people in different organisations as they interact with an electronic report of project X.

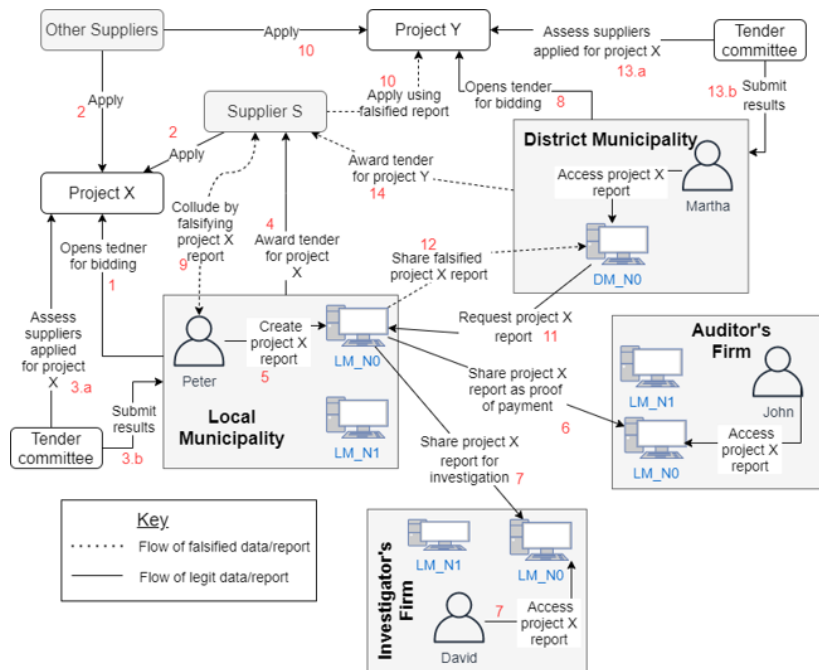


FIGURE 9: SCENARIO

To support the CTS, this study proposed a distributed model, instead of conventional email, that seeks to connect all the computers of various organisations that have an interest in the tendering project. For instance, the computers that have an interest in project X are LM_N0, DM_N0, IF_N0 (Investigator's Firm node 0), and AF_N0 (Auditor's Firm node 0) as shown in Figure 9. Therefore, the proposed model would be used as a tool that replaces email when it comes to sharing project information with all the people that have an interest in the tendering project. Additionally, the establishment of the Blockchain network also allows these computers to share project information securely while preserving the integrity of the information. The establishment of the ShareTendPro network as a solution is also aimed at enforcing trust and transparency among various organisations that have an interest in the tendering project.

Figure 10 depicts how this study addresses the identified problem within the scenario by introducing the ShareTendPro network as a solution. A more detailed discussion of the ShareTendPro solution as shown in Figure 10 follows next to solve the problem shown in Figure 9.

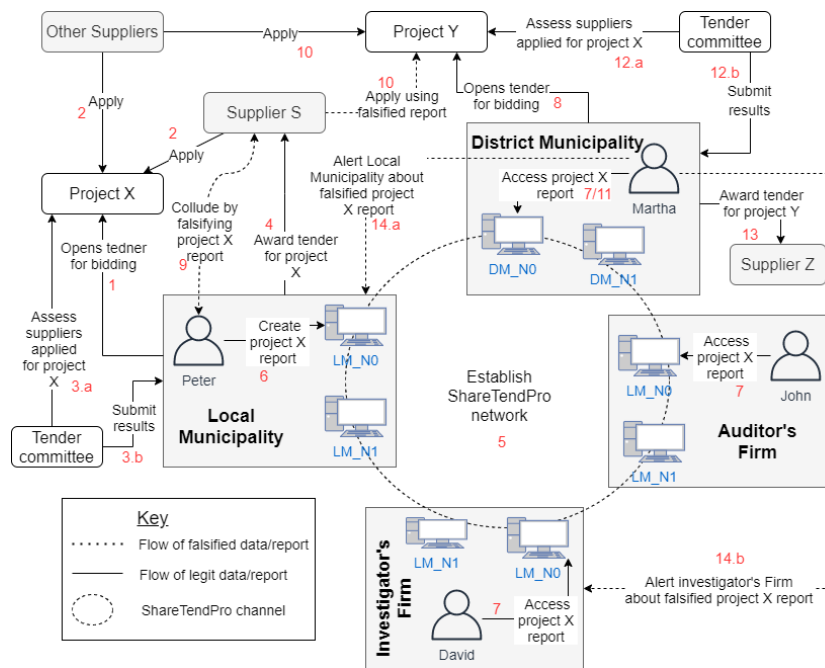


FIGURE 10: SHARETENDPRO SOLUTION

The process taking place within the ShareTendPro model is as follows:

1. Steps 1-4: these steps are similar to steps 1-4 as discussed in the scenario of Figure 9.
5. Step 5: represents the establishment of the ShareTendPro network that would be used to share project information securely while preserving the integrity of the information.
6. Step 6: depicts Peter using computer LM_N0 to create a progress report of project X. Note that computer LM_N0 is one of the computers of the LM that has joined the ShareTendPro network – hence, the report created by Peter would be stored within the blockchain of the ShareTendPro network.
7. Step 7: depicts various computers accessing the report of project X that was created using computer LM_N0. Note that this step is automatically activated when computer LM_N0 submits the report of project X to the Blockchain network of the ShareTendPro model, whereby the ShareTendPro network distributes it to all the computers that have joined the communication channel, due to the inner workings of the Blockchain.
8. Step 8: depicts the DM opening project Y for bidding. This step is similar to step 8 of Figure 9.
9. Step 9: depicts Supplier S and Peter colluding by falsifying the report of project X. This step is similar to step 9 of Figure 9. Later (in step 12) it will become clear how this falsification is detected.

10. Step 10: depicts various suppliers applying for tendering project Y offered by the DM, including Supplier S. This step is similar to step 10 of Figure 9. Assume that Supplier S has included the falsified report on the tendering documents when bidding for project Y.
11. Step 11: represents Martha who was tasked by the tendering committee of project Y to confirm the progress report submitted by Supplier S within the ShareTendPro network. Note that Martha at node DM_N0 did not request the report of project X as compared to the scenario depicted in step 11 of Figure 9 because the report is now available in the ShareTendPro network (Blockchain) as she can access it directly.
12. Step 12: depicts the tendering committee of the DM assessing all the suppliers that have applied for project Y and submitting the results of the assessment to the DM. However, the tendering committee realised that the report (i.e. document) of project X submitted by Supplier S contradicts the actual details (i.e. the report) stored within the blockchain of the ShareTendPro network. Due to this discrepancy, Supplier S is removed from the bidding process of project Y with consequences, and another supplier will need to be appointed.
13. Step 13: depicts the DM awarding tender project Y to Supplier Z. Note that this was achieved after penalising Supplier S since the information or report provided by the supplier does not correspond with the actual report stored within the ShareTendPro network.
14. Step 14: represents Martha who is part of the tendering committee of project Y alerting the LM and IF about the falsified report of project X for further investigations. The LM will conduct an internal investigation to discipline Peter, while the IF will conduct corruption-related activities or investigations between Peter and Supplier S which include acts of bribery. This, however, is out of the scope of this research and will not be shown further.

VI. Related work

There are several related works that can be associated with this study. However, some of them tend to focus more on the procurement processes, which includes processes such as applying for a tender, submitting tender documents, tender bidding, and awarding of tenders [17][18][19][20], including managing tender contracts or construction projects [21][22][23]. For instance, the following studies [18][24][25] can be associated with the procurement processes because they contain some of the elements that are related to tender bidding.

The framework presented by [22] focused on how the Blockchain can be used to facilitate data integrity within the document management for construction projects, while the study done by [26] also proposes a model that focuses on managing tendering contracts. The Mexican Government also implemented a similar tool that seeks to manage the contract of their procurement processes [27]. This study has noted that most of these proposed concepts or prototypes make use of the Ethereum platform to achieve their desired objectives, while some of them use a deprecated tool called Hyperledger-composer (HLC). The Ethereum platform relies on miners to add new transactions to the network, and it also uses the native cryptocurrency called Ether [28]. The HLC tool is regarded as a deprecated tool because none of its maintainers are actively providing support or developing new features on it [29].

All these studies tend to share tendering project information with a limited number of parties, especially parties that are involved in the procurement processes. A study done by [30] presented an open government concept that seeks to promote transparency within the procurement processes and the importance of sharing project information with various parties that have an interest in it. The following study [23] proposed a framework that might be adopted by the South African government to reduce corruption and other issues that emanates from managing procurement contracts. However, this study took a slightly different approach since it proposes a concept that can be used to monitor the tendering project, including sharing project information securely and efficiently among various parties that have an interest in the tendering project. Therefore, Table 1 depicts the comparison of the related works and the ShareTendPro model. Note that the comparison is based on the features or potential benefits offered by the adopted technology solution.

Table 1: compares related work with the ShareTendPro

Features or potential benefits	[17, 18, 20, 21, 22, 25, 30, 31]	[23]	[24]	[26]	ShareTendPro
Support private Blockchain		√	√		√
Support smart contract	√	√	√	√	√
Share tender information	√	√	√	√	√
Does not use deprecated tool	√	√			√
Does not use cryptocurrency			√		√
Not used for tender bidding				√	√
Does not use mining			√		√

As indicated in Table 1, the ShareTendPro model met all the features or potential benefits offered by the adopted technology solution. However, only two of the related work support the configuration of a private Blockchain network, which implies that others are configured for either public Blockchain or public-permissioned Blockchain. Note that a public Blockchain network enables anyone to join and participate in the network, while a private Blockchain allows only selected actors to participate in it. All the related work supports the use of SC as a mechanism to govern their transactions with an aim of securely sharing project information. Two of the related work rely on a deprecated tool, which is HLC. One of the related works does not rely on either cryptocurrency or mining algorithms to add new transactions to the network. Lastly, one of the related works does not support tender bidding processes since it focuses on managing tender contracts as indicated earlier on.

This study acknowledges that a full experimental evaluation of the CTS and proposed solution was not conducted due to time constraints. Hence, the security comparison of the existing system and the proposed solution were not fully detailed. However, this study makes use of the potential benefits associated with adopted technology solution to determine the security aspects of the proposed solution. For instance, the proposed solution is more secured since its data is distributed in multiple locations or different organisations that use different security mechanisms to secure their data, unlike the existing systems whereby a particular organisation is responsible for securing its data. Additionally, this mechanism of sharing data makes it difficult for unauthorised parties to compromise the project information once it has been stored within the network since it requires them to simultaneously hack all the organisations that form part of the network to compromise or access the data stored in it. Furthermore, the proposed solution is more secured because it uses various security mechanisms such as cryptography, timestamp, and distributed ledger, as well as having immutable data.

One of the main foreseeable shortcomings that might arise is the lack of political will to adopt the proposed solution because most of high-ranking positions within these institutions are influenced by political ideology. Hence, they might exist some reluctance when it comes to adopting a solution that seeks to reduce issues that emanate from corruption within the tendering system.

VII. Conclusion

The proposed model demonstrates how DLT can be used to share project information securely and efficiently with all the parties that have an interest in the tendering project. Some of the information security mechanisms used by the adopted technology are DLS, cryptographic encryption techniques, and having immutable data or transactions. The proposed model seeks to promote the need for sharing project information while eliminating issues that are related to a single point of failure or having an organisation that has central powers over project information. Additionally, the adoption of DLT incorporates the benefits and promises that come with this new technology. The SALG uses tendering projects to promote collaboration between public and private sectors, therefore, the proposed model becomes an essential platform that can be used to securely share project information without colluding.

In the future, this research will focus on the design and implementation of the proposed model as part of trying to come up with the proof of concept related to sharing of project information among all the parties that have an interest in it.

Acknowledgment

This research study was funded and supported by the Council for Scientific and Industrial Research (CSIR) and the University of Pretoria (UP). Special thanks go to Prof. H. Venter (UP) and Mr. H. Le Roux (CSIR) for their continuous support and contribution towards the success of this research.

References

- [1] P. Bolton, "Government procurement as a policy tool in South Africa," *Journal of Public Procurement*, vol. 6, no. 3, 2006.
- [2] C. Waters and D. Waters, "Operations management: producing goods and services," in *Pearson Education*, 2002.
- [3] S. Ngobeni, "An analysis of the tender process in national government in South Africa," in *MBA Thesis, North-West University, Potchefstroom Campus*, Potchefstroom, 2011.
- [4] National Treasury, "Legislation: PFMA-Supply Chain Management: General Procurement Guidelines," [Online]. Available: <http://www.treasury.gov.za/legislation/pfma/supplychain/General%20Procurement%20Guidelines.pdf>. [Accessed 13 12 2018].
- [5] P. Munzhedzi, "South African public sector procurement and corruption: Inseparable twins?," *Journal of Transport and Supply Chain Management*, vol. 10, no. 1, pp. 1-8, 2016.
- [6] Council of Supply Chain Management Professionals (CSCMP), "Certify," CSCMP Fundamentals, [Online]. Available:

- https://cscmp.org/CSCMP/Certify/Fundamentals/What_is_Supply_Chain_Management.aspx. [Accessed 06 11 2018].
- [7] South African Government, "Documents: The Constitution of the Republic of South Africa - Chapter 13: 213-230 Finance," Constitution, 1996. [Online]. Available: <https://www.gov.za/documents/constitution-republic-south-africa-1996>. [Accessed 13 12 2018].
- [8] South African Government, "Preferential Procurement Policy Framework Act 5 of 2000," Republic of South Africa, 2018. [Online]. Available: <https://www.gov.za/documents/preferential-procurement-policy-framework-act>. [Accessed 13 12 2018].
- [9] National Treasury, "Legislation: Local Government- Municipal Finance Management Act, No. 56 of 2003," 2003. [Online]. Available: <http://mfma.treasury.gov.za/Legislation/lgmfma/Pages/default.aspx>. [Accessed 13 12 2018].
- [10] South African Government, "Documents - Acts: Public Finance Management Amendment Act," 1999. [Online]. Available: <https://www.gov.za/documents/public-finance-management-amendment-act>. [Accessed 14 12 2018].
- [11] National Treasury, "MFMA Guidelines: Supply Chain Management," 10 2005. [Online]. Available: http://mfma.treasury.gov.za/MFMA/Guidelines/Guide%20for%20Municipal%20Accounting%20Officers_1.pdf. [Accessed 24 10 2018].
- [12] F. Otieno, "The roles of monitoring and evaluation in projects," in *In 2 nd International Conference on Construction in Developing Countries: Challenges facing the construction industry in developing countries*, 2000.
- [13] F. Alessio and P. Pythagoras, "Blockchain, Enterprise Resource Planning (ERP) and Accounting Information Systems (AIS): Research on e-Procurement and System Integration," *Applied Sciences*, 2021.
- [14] ASTRI, "Whitepaper on Distributed Ledger Technology," 11 November 2016. [Online]. Available: <https://www.astri.org/tdprojects/whitepaper-on-distributed-ledger-technology/>. [Accessed 18 02 2019].
- [15] H. Natarajan , S. Krause and H. Gradstein , "Distributed Ledger Technology and Blockchain," The World Bank Group: Open knowledge Repository, 2017.
- [16] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," www.bitcoin.org, 2008.
- [17] D. Mali, D. Mogaveera, P. Kitawat and M. Jawwad, "Blockchain-based e-tendering system," in *2020 4th International Conference on Intelligent Computing and Control Systems*, pp. 357-362, 2020.
- [18] X. Li, "BCES: A BlockChain based Credible E-Bidding System," in *IEEE 6th International Conference on Computer and Communications*, 2020.
- [19] J. Deshpande, M. Gowda, M. Dixit, M. Khubbar, B. Jayasri and S. Lokesh, "Permissioned blockchain based public procurement system," *Journal of Physics: Conference Series*, vol. 1706, no. 1, 2020.
- [20] T. Weingärtner, D. Batista, S. Köchli and G. Voutat, "Prototyping a Smart Contract Based Public Procurement to Fight Corruption.," *Computers*, vol. 10, no. 7, p. 85, 2021.
- [21] I. Omar, R. Jayaraman, M. Debe, K. Salah, I. Yaqoob and M. Omar, "Automating procurement contracts in the healthcare supply chain using blockchain smart contracts," in *EEE Access*, 9, 2021.

- [22] M. Das, X. Tao, Y. Liu and J. Cheng, "A blockchain-based integrated document management framework for construction applications," in *Automation in Construction*, 133, 104001, 2022.
- [23] O. Ogunlela, O. Ojugbele and R. Tengeh, "Blockchain technology as a panacea for procurement corruption in digital era," *International Journal of Research in Business and Social Science*, vol. 10, no. 4, 2021.
- [24] Y. Goswami, A. Agrawal and A. Bhatia, "E-Governance: A Tendering Framework Using Blockchain with Active Participation of Citizens," In *2020 IEEE International Conference on Advanced Networks and Telecommunications Systems*, 12 2020.
- [25] V. Hassija, V. Chamola, D. Krishna, N. Kumar and M. Guizani, "A blockchain and edge-computing-based secure framework for government tender allocation," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2409-2418, 2020.
- [26] S. Perera, S. Nanayakkara, M. Rodrigo, S. Senaratne and R. Weinand, "Blockchain technology: Is it hype or real in the construction industry?," *Journal of Industrial Information Integration*, vol. 17, p. 100125, 2020.
- [27] F. Zbinden and G. Kondova, "Economic development in Mexico and the role of blockchain," *Advances in Economics and Business*, vol. 7, no. 1, pp. 55-64, 2019.
- [28] Ethereum, "What is Ethereum?," Ethereum.org, [Online]. Available: <https://ethereum.org/en/what-is-ethereum/>. [Accessed 02 11 2021].
- [29] Linux Foundation Projects, "Hyperledger-composer," Hyperledger Foundation, August 2021. [Online]. Available: <https://www.hyperledger.org/use/composer>. [Accessed 16 05 2022].
- [30] F. Hardwick, R. Akram and K. Markantonakis, "Fair and Transparent Blockchain Based Tendering Framework - A Step Towards Open Governance," *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering*, 2018.
- [31] D. Čeke, N. Buzadija and S. Kunosić, "Enhancing transparency and fairness in public procurement process with the support of blockchain technology: a smart contract based approach," *21st International Symposium INFOTEH-JAHORINA*, March 2022.

D.3. Using distributed ledger technology for digital forensic investigation purposes on tendering projects. International Journal of Information Technology (IJIT)

Abstract

The South African Local Government (SALG) uses the tendering system to procure goods and services. Some of these tendering projects are aimed at promoting socio-economic and industrial policies. Hence, the tendering system used by SALG should be fair, transparent, competitive, cost-effective, equitable, and free from corruption. However, the mismanagement of the tendering system might lead to interruption of operations, late service delivery, rising costs, and most importantly, fraud and corruption. The use of paperwork to share project information might lead to the mismanagement of the tendering project because it might contribute towards illicit altering of project information during the process. The purpose of this study is to develop a Blockchain prototype that might be used to securely share project information with all the parties interested in the tendering project. It is recommended that the adoption of the proposed solution will enable various organisations to have access to real-time data, allowing them to have access to the entire project history regardless of their geographical location. Access to real-time data would promote real-time auditing and digital forensic investigations because both auditors and investigators will have access to credible digital evidence or project information of their interest in real-time.

Keywords—Tendering system, Blockchain, project information, sharing tendering projects, South African Local Government.

I. Introduction

We are living in a digital world where most people are exposed to digital information. This digital information is driven by the advancement of technologies that are being used daily. Digital information requires electronic devices that have the capabilities of processing and manipulating digital data. These capabilities tend to affect the perception of society as they rely in some way on how we collect, process, analyse and retrieve data more easily and efficiently. Some of these capabilities also play an important role when it comes to new innovations in information and communication technologies (ICTs). The evolution of these technologies has positively benefited society by providing access to information and improving communication channels. The internet is one of the most used platforms that provide such services. Adversely,

various risks come with the usage of the internet. These risks include identity theft, cybercrime, fraud, and many other malicious activities [1][2]. The most used electronic devices that are targeted by these activities are computers and mobile devices. However, these cyber threats do not stop the adoption of ICTs as a tool that aims at enhancing the standard of living [3], because as ICTs evolve, new mechanisms are being implemented to address some of these risks.

The South African organs of state have also adopted the use of these devices as a tool that enables them to perform some of their tasks. Even though some of their tasks are still using manual processes, where they rely heavily on paperwork to accomplish certain tasks. For instance, various South African organs of state collect tendering information through the use of paperwork, which will then be captured and converted into a digital format. Tendering is one of the methods used by these organs to deliver some of their basic services. Tendering can also be regarded as a process that enables the government to procure goods and services from a contractor, which is an organisation within the private sector in most cases. Tendering data is collected whenever there is a call for tender projects. During the tendering process, there might exist some negative activities or irregularities that seek to undermine the norms of a tendering system. The following section explores this problem statement in more detail.

A. Problem statement

The current South African tendering system still relies heavily on manual processes, which requires skilled personnel to deal with manual forms and also administer the entire process [4]. The main reason behind using paperwork is to accommodate all the participants including small and new contractors because some of them are unable to share their projects due to various reasons. Some of the reasons are; lack of internet access or personal computers, which lead to the usage of files to store their project information. Project information plays an important role when it comes to awarding a tender to a particular contractor since it reflects the competency area and project history of a particular contractor. All contractors are required to submit such information when they are applying for a tender. Some of this information will go through a verification process, whereby a referee will then be contacted regarding a specific item indicated on the documents. A referee, in this case, might be either a client of that particular contractor or someone who might provide more information or clarity. Some of the tools that are used to share project information are reports, meetings, presentations, site visits, and other intermediaries such as trusted third parties. Through these processes, information might be altered for corrupt purposes at any given stage.

Therefore, the primary problem of this study is that paperwork is used to share project information, which might contribute to the illicit altering of information during the process. This might also affect the fairness, transparency, data integrity, and competitiveness of the tendering system. To provide a solution to this problem, the following questions are also addressed:

- **Research question (RQ) 1:** How does the tendering system work in the South African context?
- **RQ 2:** Is distributed ledger technology (DLT) a possible solution to the identified problem?
- **RQ 3:** How does transparency, accountability, and integrity of data in a potential solution work and how will it contribute to digital forensics?

B. Research objectives

There are various ways of providing a solution to a problem, however, there are certain goals that need to be set before attempting to solve a particular problem. Therefore, this study addresses the following objectives:

- a) To investigate how Blockchain as a technology work and how data transparency and accountability are achieved. Blockchain technology (BCT) can be viewed as peer-to-peer decentralised, DLT that is replicated to all nodes participating in a network [1].
- b) To develop a Blockchain prototype that allows various organisations to share project information securely and efficiently. The proposed prototype might also be used to improve the current tendering project communication in South Africa.
- c) To investigate how digital forensics might be applied to trace the accountability of the records or data.

The remainder of this study is structured as follows: the adopted research method is detailed in Section II. A brief overview of the background details related to the tendering system landscape and the adopted technology description is detailed in Section III. Section IV provides the details of the proposed model used to distribute project information. Section V explores the design of the proposed model. The application of the proposed model is demonstrated in Section VI, whereby a fictional use-case scenario is provided. The details of the ShareTendPro model results are discussed in Section VII. Thereafter, Section VIII depicts the evaluation of the research study by outlining some of the benefits and shortcomings associated with the adoption of the proposed solution. Finally, the last section, which is Section IX, provides the conclusion of this study, which includes a recall of the problem statement and future work.

II. Research method

This study has adopted various research methods to achieve the desired objectives, namely: design science research (DSR), literature review (LR), modelling, theoretical use-case, and prototyping. This study adopted the DSR because it is a problem-solving paradigm that seeks to develop or enhance an artifact with an aim of improving the functional performance of the existing tendering system [5]. In order to understand the identified problem, a brief LR on how the tendering system work in the SALG is detailed, which also includes how the adopted technology work in general. This study makes use of various materials to achieve this, and some of the primary and secondary materials include the South African legislation, government documents, journal articles, conference papers, as well as general internet searches. After obtaining a holistic idea of how the tendering system work, then this study proposes a model that might be used to share project information securely and efficiently with all the parties that have an interest in the tendering project. Hence, the modelling methodology was adopted to model the proposed solution. The proposed model explores how various components interact with the project information to achieve the desired objective.

The Unified Modelling Language (UML) diagrams are used to visualise how the proposed model can be used to address the identified problem. Note that this study makes use of the following UML diagrams namely use-case, state, and information flow. Additionally, this study also uses a theoretical use-case to expand the idea behind the proposed model, which is based on a fictional use-case scenario. Lastly, the prototype methodology was adopted to implement the proposed solution to provide a proof of concept³.

III. Background

The literature review contained within this section is classified into three subsections namely: the *current tendering system used by SALG*, the *adopted technology description*, and *related work*. The *current tendering system used by the SALG* focuses on the background details related to the tendering system used by the SALG. The *adopted technology description* focuses on the background details that seek to explore the technology solution adopted by this study to implement the prototype of the proposed solution, while the *related work* focuses on some of the related work that can be associated with this study.

³ Note that the prototype processes adopted during the implementation of the proposed solution are similar to the ones outlined by **Invalid source specified**.

A. The current tendering system used by SALG

The delineation of this study lies in sharing tendering project information amongst organs of state that fall within the SALG because it is the smallest sphere used by the South African Government to deliver some of the basic services to the surrounding communities. Additionally, most of these services or projects have a direct impact on the surrounding communities. The SALG is divided into three categories namely: Metropolitan (Metro), District, and Local Municipalities. District and Local municipalities share their responsibilities when it comes to executing some of the projects, while Metros are regarded as standalone municipalities since they report directly to the Provincial Government.

The Local Municipalities (LMs) are responsible for all the tendering projects that fall under their mandate, even though some of these projects are overseen by their District Municipality (DM). The execution of these tendering projects requires municipalities to contract suppliers which are capable of executing similar projects. Thereafter, all these municipalities are also required to share some of their project information with the affected parties, such as *Communities* and *Investigators*. *Communities* act as the beneficiaries of some of these projects, while the *Investigators* are responsible for investigating irregularities that might occur during the execution of some of these projects. Additionally, these municipalities are also required to share their financial reports of these projects with their *Auditors* because there are responsible for overseeing how these municipalities use public funds. The communication channel used by these municipalities to share project information is structured in a centralised manner whereby municipalities are seen as the centre that distributes project information to all the parties that have an interest in the project information, as shown in Figure 1. Furthermore, this communication channel relies heavily on paperwork to share project information, even though some of this information is used for decision making-purpose, especially when it comes to awarding a tender to a particular supplier. Therefore, Figure 1 seeks to summarise this concept by providing a high-level visualisation of how the current project information-sharing work.

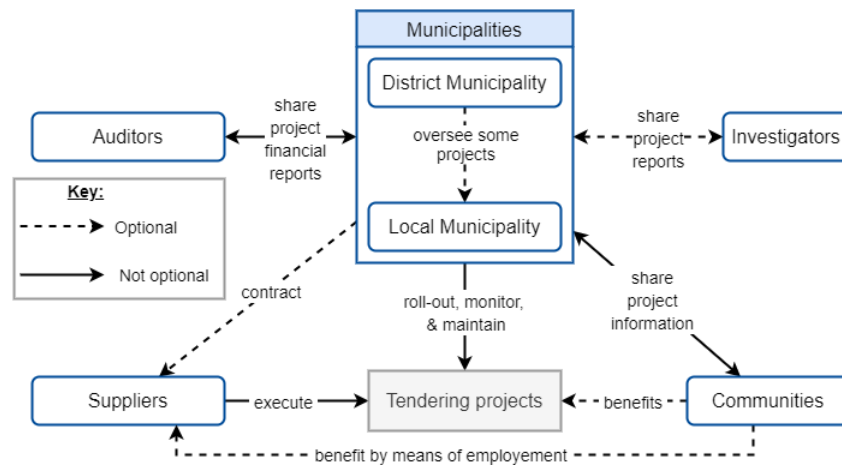


FIGURE 1: CURRENT PROJECT INFORMATION SHARING CONCEPT

This section has outlined an overview of how the tendering system works in the SALG, which includes identifying various stakeholders (i.e., Auditors, Investigators, Communities, Suppliers, District and Local Municipalities). Therefore, the following section seeks to explore the description of the adopted technology that might be used to share project information securely and efficiently.

B. Adopted technology description

This study adopts DLT as the technology solution that might be used to address the identified problem. ASTRI [6] defined DLT as a “technology protocol that can be used for developing a replicated and shared ledger system that stores a wide range of assets and transactions in a distributed manner”. This implies that a distributed ledger (DL) is regarded as a shared ledger system since its records of transactions are maintained across several locations or among multiple nodes, regardless of their geographical location [7]. Furthermore, it implies that all the nodes that are found within that network have the same copy of the ledger. Hence, a DL does not consist of a central repository or a single point of failure like a centralised ledger system. However, every time when a specific node in a DL has made some valid changes on the ledger, those changes are propagated automatically and shared with other nodes that form part of the network. Additionally, this mechanism of sharing information ensures that there is no single point of failure and it is also aimed at maintaining data integrity across all the nodes within that network.

Note that the DLT has become more prevalent in 2008, after the circulation of a white paper titled “Bitcoin: A Peer-to-Peer Electronic Cash System” authored by Satoshi Nakamoto [8]. The white paper proposed a solution for the financial industry that addresses the issue of double-spending and eliminating the norm of using intermediaries. However, the ideology of the

proposed solution existed theoretically [9][10], until 2009 when the first DLT implementation (Bitcoin system) emerged [8]. The underlying technology used to implement the Bitcoin system was termed “Blockchain” technology. Blockchain refers to the ways in which the proposed system stores and organises its information. The word “Blockchain” is a combination of two words namely “block” and “chain”. This is due to the fact that “Blockchain” technology use blocks to store their information, and these blocks are linked together to form a chain-like data structure, hence “Blockchain”. As time progresses, similar ways of organising and storing information emerged which led to the term DLTs as a broad term used to categorise such technologies [7].

Various DLTs support or use the Blockchain data structure to organise and share their information. However, all the frameworks that adopt the use of Blockchain data structure can be classified into three categories namely [11]:

- *Permissionless (public) Blockchain* allows any member of the public to join the network and participate in it.
- *Public-permissioned Blockchain* allows any member of the public to verify the records or transactions stored in the network.
- *Private-permissioned (private) Blockchain* allows specific members to participate in the network, hence it is designed to support private network configurations.

There are various Blockchain frameworks that might be adopted to implement a distributed solution that seeks to address the identified problem. Therefore, the following subsections explore the most popular Blockchain frameworks which are *Bitcoin*, *Ripple*, *Ethereum*, and *Hyperledger-fabric* (HLF) [6].

1. The *Bitcoin framework* is the first DLT implementation that was specifically designed to support the native cryptocurrency known as Bitcoin [11]. Additionally, this framework is regarded as a public Blockchain that uses a mining consensus algorithm called proof-of-work [12]. Furthermore, it relies on miners to add new transactions to the network.
2. The *Ripple framework* is specially designed for digital currency exchange, remittance, and the real-time gross settlement system (RTGS) [13]. The RTGS is an open-source, distributed technology that focuses on payment systems, particularly in banking and finance [14]. It supports a native cryptocurrency known as Ripple and it also uses a custom-made consensus algorithm called the ripple protocol consensus algorithm [15][16].
3. The *Ethereum framework* is specially designed to support the native cryptocurrency known as Ether [17]. It also supports smart-contract (SC) [18], which is the mechanism used by

some of the DLTs to govern the network transactions, without relying on a trusted party or authority to mitigate the transaction processes. However, the SC used by Ethereum is written in high-level languages (e.g., solidity [19]) and compiled by bytecode which requires an Ethereum Virtual Machine (EVM) to execute it [20][21]. Furthermore, the Ethereum framework can be regarded as both a private and a public Blockchain that uses proof-of-work and proof-of-stake consensus algorithms [22].

4. The *Hyperledger-Fabric framework* is an open-source platform hosted by Linux Foundation, created to advance cross-industry Blockchain solutions [23]. It does not support native cryptocurrency since it was designed to build a new generation of transactional applications that aimed at establishing trust, accountability, and transparency [23]. Additionally, it supports private Blockchain and uses a crash fault-tolerance consensus algorithm [24]. HLF also supports the use of SC (also known as chaincode), which can be viewed as a mechanism that seeks to manage access and modification of the data within the network.

5. *Selecting a particular Blockchain framework*

Table 1 compares the above Blockchain frameworks to select a suitable framework that might be adopted by this study. The comparison is based on whether the identified requirements are favourable or not. However, some of these requirements are based on the features or benefits offered by these frameworks.

TABLE 3: COMPARISON OF BLOCKCHAIN FRAMEWORKS

Requirements	Blockchain framework [6][25][26]			
	Bitcoin [8]	Ripple [14]	Ethereum [27, 28]	HLF [27, 29, 30]
Support cross-industry application development			√	√
It must not rely on a native cryptocurrency				√
Support private Blockchain configurations		√	√	√
Support SC			√	√
It must not rely on miners to add new transactions		√		√
It must support a data auditing mechanism		√		√
Support Blockchain data structure	√	√	√	√

The comparison of these Blockchain frameworks favours HLF because all the requirements are met. Additionally, all the participating members of HLF are also known (since it is a private Blockchain) and such members are, therefore, accountable for their actions. Hence, it can be assumed that all the participants can be trusted with the assigned tasks. Note that the participating members, in this case, refers to the identified stakeholder in the previous section, which are Auditors, Investigators, Communities, Suppliers, District, and Local Municipalities.

This section has highlighted the background details of the adopted technology, which included the selection of HLF as a favourable Blockchain framework. Therefore, the following section focuses on the related work that can be associated with this study.

C. Related work

There are several related works that can be associated with this study. Some of these related works tend to focus more on the procurement processes, which includes processes such as applying for a tender, submitting tender documents, tender bidding, and awarding of tenders,

including managing tender contracts or projects. Various studies classify the following processes: applying for a tender, submitting tender documents, tender bidding, and awarding of tenders as e-procurement because their information is widely used during the procurement proceedings. The management of the tendering contract or projects tends to come after the e-procurement processes to combat issues that emanate from duplication of contracts or tendering projects.

For instance, the study done by [31] proposed an e-procurement system that can be used to create, publish, bid, and award tendering projects. The proposed system is based on the Ethereum platform, which relies on cryptocurrency or mining algorithms to add new transactions to the main network. The study by [32] also adopted the Ethereum platform to expand the tender bidding concept by including processes such as sharing and verifying tendering information. Additionally, the study done by [33] also adopted a similar approach to expand the tender bidding concept by including processes such as supplier habilitation and delivery verification. However, the model presented by [34] has adopted a different approach or technology solution since it uses the Hyperledger-composer (HLC) tool to implement a prototype that can be used to share data associated with the bidding and awarding of tender projects. Note that HLC makes use of HLF as the underlying Blockchain framework. Additionally, HLC is also regarded as a deprecated tool because none of its maintainers are actively providing support or developing new features for it [35]. The solution presented by [36] also adopted HLF to expand the tender bidding concept by including a mechanism that can be used to monitor the procurement proceedings.

The following studies [37][38] can be associated with managing tender contracts because they contain some of the elements that seek to eliminate issues that emanate from duplication of contracts or projects, especially in the public sector. Some of the issues that are addressed by these studies relate to data integrity, transparency, and accountability among various individuals that are involved in finalising the procurement contracts. The Mexican Government is one of the countries that has implemented a tool that seeks to manage its procurement contracts [39], especially managing contracts of the projects that are executed using tendering systems.

The framework presented by [40] focused on how the Blockchain can be used to facilitate data integrity within the document management for construction-related projects. Note that the proposed framework is also based on the Ethereum platform. Additionally, the work presented by [41] explores a framework that can be used to secure tendering records that are highly susceptible to tampering. The study conducted by [42] expanded this ideology by including a

concept that seeks to manage construction projects executed by multiple constructors to provide transparency and accountability within the project.

All these studies tend to share tendering project information with a limited number of parties, especially parties that are involved in the procurement processes. A study conducted by [43] presented an open government concept that seeks to promote transparency within the procurement processes and the importance of sharing project information with various parties that have an interest in it. The study by [44] proposed a framework that might be adopted by the South African Government to reduce corruption and other issues that emanates from managing procurement contracts. However, this study took a slightly different approach since it proposes a concept that can be used to monitor the tendering project, including sharing project information securely and efficiently among various parties that have an interest in the tendering project.

Table 2 summarises the details of the related work by providing a comparative survey that seeks to outline some of the features or issues that were not addressed by these related works. As indicated in Table 2, most of the related work make use of the Ethereum platform as their technology solution, while this study adopted the use of HLF. It should be noted that the features in the last four columns of Table 2 resemble positive features. For example, the column on “Does not support tender bidding” should be conceived as positive because this study focuses on monitoring the execution of tendering projects rather than processes that fall within e-procurement. The notion of monitoring tendering projects aimed at ensuring that it is executed successfully and all the parties that are involved during the execution phase account for their action.

Table 2: A comparative survey of the related work

Ref.	Key contribution	Features or issues					
		Industry / department	Blockchain Technology	Does not support tender bidding	Support the execution of projects	Does not rely on mining algorithms	Support private Blockchain
[79]	e-tendering system (create, publish, bid, evaluate, & award tender)	Supply chain	Ethereum				
[80]	e-bidding system (sharing & verifying data)	Supply chain	Ethereum				
[84]	Tender bidding and monitoring framework	Supply chain	HLF		√	√	√
[81]	Bidding process, supplier habilitation & delivery verification	Supply chain	Ethereum		√		
[85]	Contract management	Healthcare, Supply chain	Ethereum	√			
[92]	Public procurement framework (contract management)	Supply chain	N/A	√			
[82]	Tendering system (sharing tender data, bidding, & awarding tender)	Supply chain	HLF, HLC			√	√
[89]	Government tender framework	Construction	Ethereum	√	√		
[90]	Managing construction projects	Construction	Ethereum	√	√		
[91]	Government tendering process	Supply chain	Ethereum				
[86]	Contract management (tender bidding, evaluation & awarding)	Supply chain	Ethereum				
[93]	Supply chain conceptual model	Supply chain, logistics process	Ethereum	√			

IV. A forensic Blockchain model for distributing tendering project information

The proposed model is aimed at securely and efficiently distributing project information among all the parties that have an interest in the tendering project. The proposed model is called the share tendering project (ShareTendPro) model. The ShareTendPro model incorporates the following components, i.e. *actors*, *gateway*, and *Blockchain network*, to achieve the desired objective of securely sharing project information amongst various participants. Figure 2 depicts how these components of the proposed model interact with each other.

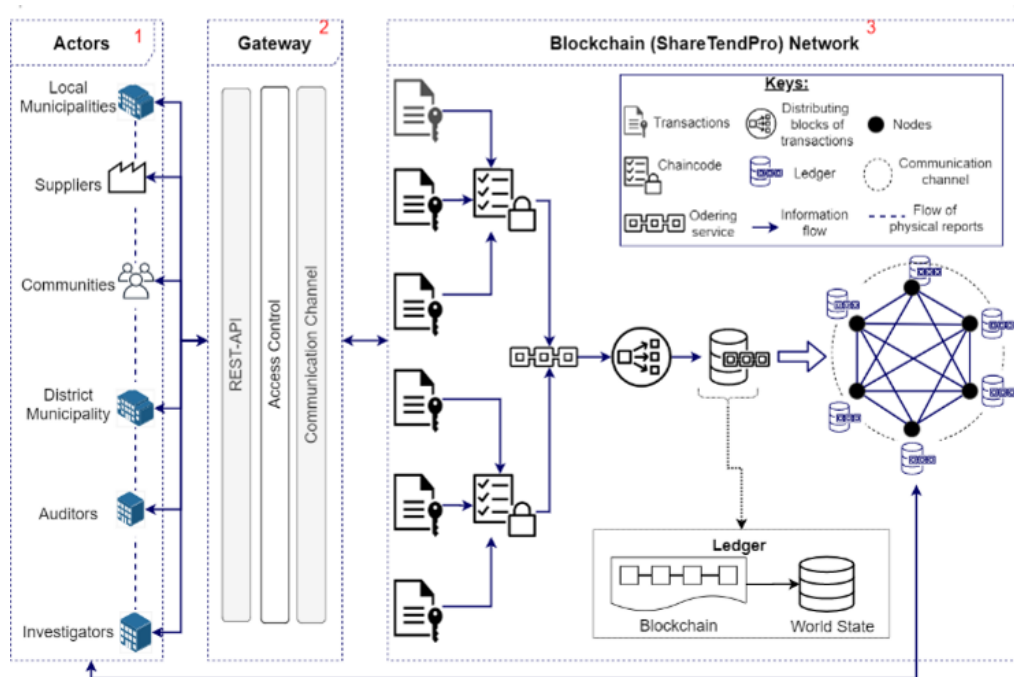


FIGURE 2: SHARETENDPRO MODEL

The following items briefly explore the logic behind these components (depicted in Figure 2) as used by the ShareTendPro model:

1. *Actors* are the role players of the proposed solution and may, for example, consist of various organisations. Various actors that were identified are Suppliers, Communities, Investigators, Auditors, DMs, and LMs.
2. The *gateway* allows actors to interact with the Blockchain network. This component seeks to manage the identities of various resources (i.e., actors and nodes), including providing access to the Blockchain network as they interact with project information. HLF achieves this by using the following mechanisms: REST-API (Restful application programming interface), access control list, and secure communication channel. REST-API allows various actors to use API (application programming interface) to interact with the Blockchain

network. Access control list manages the access rights of all the authorised resources as they interact with Blockchain data, while the secure communication channel allows a specific group of actors to secretly share project information.

3. The *Blockchain network* stores and distributes project information among all the nodes used by various actors that form part of the network.

The *Blockchain network* component is one of the main key components that enable the ShareTendPro model to achieve its objectives because it is responsible for storing and sharing project information securely and efficiently. The flow of project information within the Blockchain network component is explored in detail in Figure 2. The Blockchain network component accepts all the transactions submitted by authorised actors. These transactions are then encrypted as part of preserving data integrity and confidentiality within the network. It will then take all these transactions and assess them based on the predefined requirements associated with each transaction as stipulated on the chaincode. If these transactions meet all the requirements, then they are declared as accepted transactions. All the accepted transactions are forwarded to the ordering service to be grouped into blocks. Thereafter, all the ordered transactions will be shared with all the nodes that are found within that network. Each node will then use the chaincode to verify these ordered transactions (blocks) before they can append or add the new block to the ledger. Note that this ledger consists of two elements namely: *Blockchain data* and *world-state (WS) data* as shown in Figure 2. The information stored in the *Blockchain data* is immutable by default, while the information stored in the *WS data* changes frequently as the state of the ledger changes when new blocks are appended. Hence, the *Blockchain data* can be viewed as the audit log or evidence storage since this information cannot be changed once it has been stored or recorded (i.e. appended to the Blockchain). Thereafter, all the actors that have an interest in a specific tendering project will now have access to the project information related to that project once it has been appended successfully.

This section has explored the details of the ShareTendPro model. Therefore, the following section focuses on the design of the ShareTendPro model.

V. ShareTendPro model design

This section is classified into two subsections namely: the *requirement specifications*, and the *ShareTendPro design*. The requirement specifications seek to outline the functional and non-functional requirements, while the ShareTendPro design seeks to focus more on the design of the proposed model.

A. Requirement specifications

1. Functional requirements of the proposed solution are:

- *Adding resources* allows actors to add various resources to the Blockchain network based on the role they play. Some of these resources include tendering project information, organisations and their members.
- *Submit project reports* allows authorised actors (i.e., Investigators, Auditors, Suppliers, Communities, LMs, and DMs) to create or share project reports related to a specific tendering project.
- *Update project reports* allow authorised actors to modify the project reports stored within the Blockchain network.
- *View project reports* allow all the actors who have an interest in a specific tendering project to view all the reports related to that project.
- *Delete project information* allows authorised actors to delete project data. However, note that the evidence of this data will still exist within the Blockchain data component of the ledger as indicated earlier on.
- *Access project history* allows authorised parties to access the entire history of the tendering project.

2. Non-functional requirements

- *Confidentiality*– is achieved using cryptography since it seeks to secure project information from both internal and external threats.
- *Integrity*– having immutable data that is distributed across the network prevents unauthorised modification of project information that might be processed unnoticed.
- *Availability*– having access to the entire project history in real-time enables various parties to access the project data of their interest. Additionally, the digital forensic investigator should be able to navigate through the entire history of the tendering project effortlessly since the proposed solution has a built-in audit trail feature.
- *Scalability*– the ShareTendPro model should support a large volume of data or project information without its performance being affected.
- *Efficiency*– the efficiency of the model lies in the response time and data storage. For instance, all the project information will be distributed among all the actors that have an interest in it, unlike the current system whereby such information is shared with specific actors only, while other actors are required to submit proposal requests to access it.

- Usability– the model should be implemented in such a way that it is easy to use and operate.
- Reliability– the model should provide consistent and accurate performances based on the intended functions. All the technical errors should be minimised and assigned a meaningful error message as part of simplifying the proposed solution, including eliminating its complexity.

This section has outlined both functional and non-functional requirements of the proposed model. Therefore, the following section explores the ShareTendPro model design.

B. ShareTendPro design

The details contained within this section are classified into two categories namely: architectural and behavioural design. Hence, the following subsections explore the details of these designs starting with the architectural design, then followed by the behavioural design.

1. Architectural design

This section makes use of a three-tier architectural design, which seeks to depict the logic of the ShareTendPro model. The three-tier architectural design consists of the following layers: *interface*, *business*, and *data layers* as shown in Figure 3. The following items explore the details of these three layers:

- *Interface layer*– consists of a mechanism used by actors to interact with the data stored within the proposed model. However, note that the mechanism used by this study to interact with the proposed model is the command prompt.
- *Business layer*– focuses on the controller used to facilitate the proposed solution. The proposed model consists of various controller agents used to perform certain tasks. Some of these agents used by HLF include Docker, Docker-compose, NodeJS, Go, JavaScript, and Python.
- *Data layer*– focuses on the processes used by the proposed solution to organise and store data. The proposed solution makes use of the ledger to organise and store project information. However, this process is governed by a chaincode.

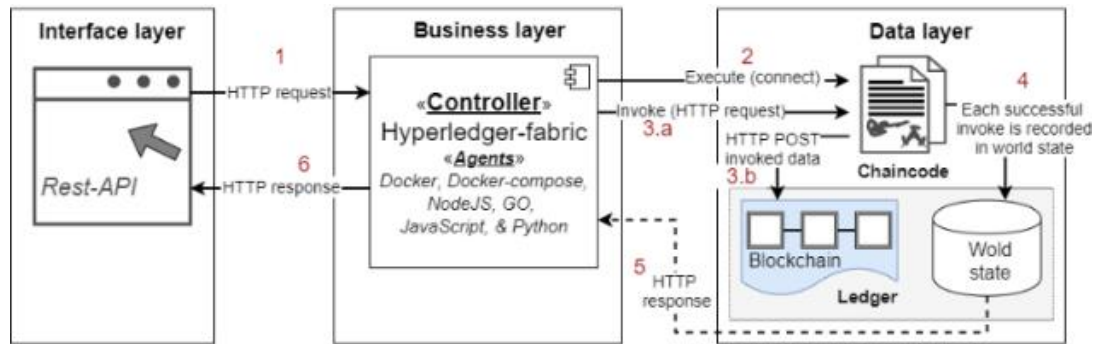


FIGURE 3: SHARETENDPRO MODEL ARCHITECTURE

Figure 3 represents the architecture of the ShareTendPro model. The numbering from 1–6 depicts the logical flow of the information as it passes through these various layers (*interface*, *business*, and *data layer*). Additionally, these layers make use of the HTTP methods to communicate with each other and these HTTP methods contain one of the following operations: POST, GET, PUT, or DELETE. The POST operation is used to store data or information in the Blockchain network. The GET operation is used to retrieve data from the Blockchain network. The PUT operation is used to update the data, while the DELETE operation is used to delete certain data from the WS within the ledger. Note that the DELETE operation only deletes information stored within the WS database, however, this information will remain immutable within the Blockchain data of the ledger as discussed in Figure 3.

This section has explored the architectural design of the proposed solution. Therefore, the following section discusses the *behavioural design* of the proposed model, which is an in-depth discussion of how information flows within the proposed solution since this section only explored the higher level of the logical information flow.

2. Behavioural design

This section explores the following UML diagrams: use-case, state, and information flow to demonstrate how actors, objects, and the environment interact with each other.

a) Use-case diagram

A use-case diagram can be viewed as a scenario-based technique used to describe all the possible actors and their interactions with the ShareTendPro model. This study has identified the following actors: DMs, LMs, Communities, Suppliers, Auditors, and Investigators as presented in the previous section. However, the actors represented in Figure 4 are reflected at a finer-grained level, i.e., where previously the actors were mainly represented on an organisational level. Note that the actors in Figure 4 are represented on the level of employees (members) working at those organisations. In addition, the Admin actor might fall under various categories

hence it is depicted in a general form as shown in Figure 4. Therefore, the Admin actors are responsible for adding resources. These resources can include organisations and their members. The LM and DM members have three main tasks assigned to them; the first task is to create a contract and assign a specific Supplier. The second task is to create a tender project and assign it to a specific contract, and the third task is to submit project reports related to a specific tender project. Additionally, the LM and DM members are allowed to view project reports of all the tendering projects of their interest. The other members consist of the following actors: Suppliers, Communities, Investigators, and Auditors. These members are all responsible for submitting project reports of all the tender projects that fall within their mandate, and they can also view other reports of the tendering projects of their interest.

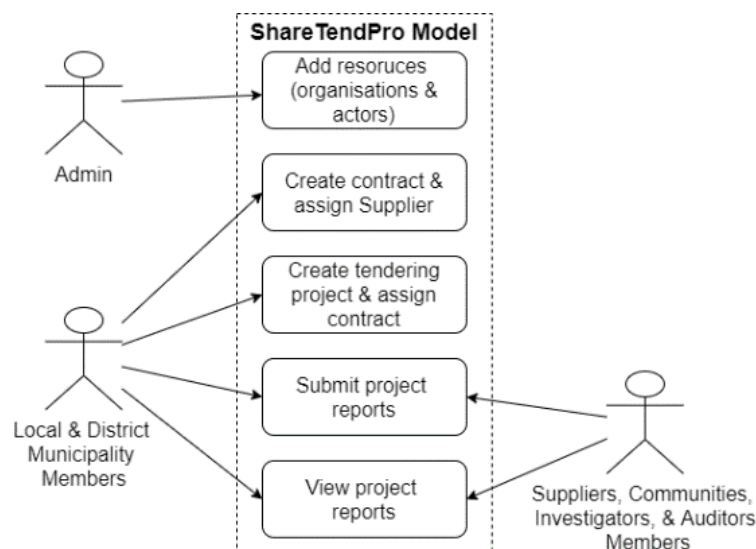


Figure 4: Use-case diagram

This section has explored the use-case diagram of the proposed solution. Therefore, the following section focuses on the state diagram of the proposed solution since it seeks to explore an in-depth information flow between various transactions submitted by these actors.

b) State diagram

Figure 5 depicts a sequence of events within the ShareTendPro model that seek to trigger the WS data as it changes from one state to the next. The numbering from T1–T6 represents the transactions within the ShareTendPro model, while label A represents the option used to verify certain information within the ShareTendPro model. The letter “T” in the numbering from T1–T6 stands for "transaction", which implies that these objects represent various transactions that seek to trigger the world state of the proposed solution. In addition, these transactions can be mapped with the use-case events highlighted in Figure 4 using the roles associated with these actors as shown in Figure 5. For instance, T1 and T2 can be associated with adding resources in

Figure 4, which reflect the admin roles. The following items explore the sequence in which these events occur:

- T1– is used for adding organisations to the network.
- T2– is used for adding members to the network and assigning them to their respective organisations.
- A– depicts an option in a form of a condition used to activate the functionalities of transactions 3 and 4, which can only be activated by either LM or DM members.
- T3– is used for creating a contract and assigning that particular contract to a specific supplier who will be responsible for executing the tendering project.
- T4– is used for creating a tender project and assigning it to a specific contract created using transaction 3.
- T5– is used for submitting project information or reports to the network.
- T6– is used for viewing project information or reports.

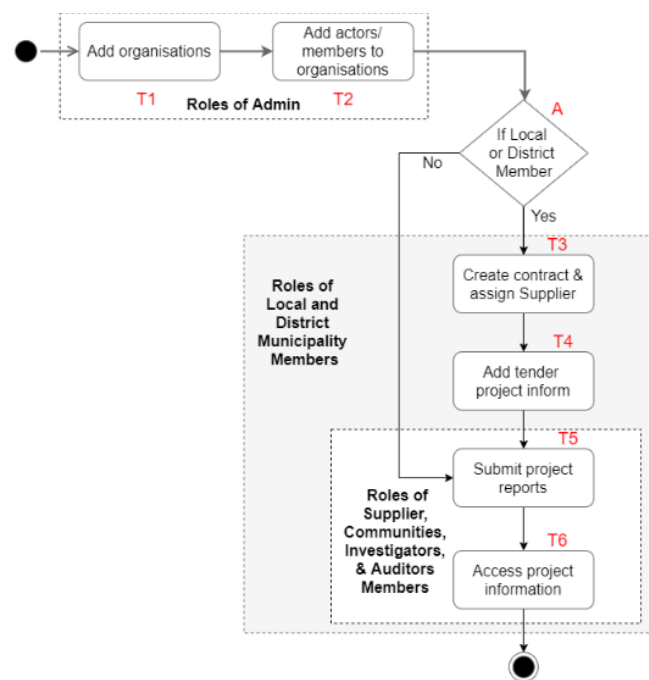


Figure5: State diagram

This section has explored the state diagram of the proposed model. Therefore, the following section focuses on the information flow that occurred between various components or objects and the environment.

c) Information flow

This section focuses on how the proposed solution work in general, which is the theoretical concept that seeks to explore the distributed nature or implementation of the proposed solution.

Additionally, the information flow discussed in this section explores the interaction between some of the embedded objects and the surrounding environment within the proposed solution. Note that HLF consists of various nodes that seek to ensure that the network achieves the desired objective of securely and efficiently distributing project information to all the nodes within the network. These nodes can be classified into three categories: *clients*, *peers*, and *orderers* [46][47]. Therefore, the following items briefly explore the details of these nodes.

- *Client nodes*– these are normal nodes or computers used by various actors to interact with the deployed network. Additionally, these nodes run an application that uses a REST-API to interact with the deployed network.
- *Peer nodes*– these nodes contain the chaincode and the ledger. The chaincode is used to govern the network by processing the transactions submitted by various actors, while the ledger is used to store the transactions submitted by various actors.
- *Orderer nodes*– these nodes are responsible for collecting all the accepted transactions within the network and grouping them into blocks. Once the grouping of these transactions is complete, then the ordered transactions (which are blocks) are shared with all the peer nodes within the network.

Figure 6 depicts a theoretical representation of the information flow within the proposed solution, including how the available resources from different organisations can be utilised to accommodate all the actors. For instance, the proposed solution will enable actors such as Suppliers and Communities to utilise the municipality’s resources whenever they want to share their project information related to a particular tendering project. Figure 6 also represents how the identified nodes interact with each other to accomplish certain objectives within the network. However, all these nodes are represented using Docker containers (virtual nodes) within this study as supported by the HLF framework. A Docker container is a component that can be used to package a code or an application with all its dependencies (i.e., libraries) and deployed as one package [48]. Figure 6 is explained in detail below.

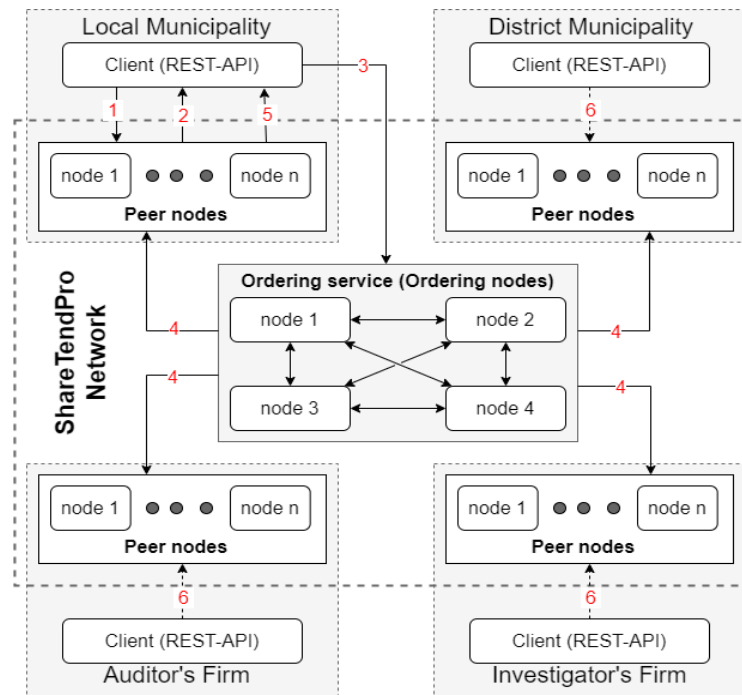


FIGURE 6: SHARETENDPRO MODEL TRANSACTION FLOW

Figure 6 consists of four organisations namely: *LM*, *DM*, *Auditors Firm*, and *Investigators Firm*. Each of these organisations contains a client node and n number of peer nodes (which are labelled from nodes 1- n). Additionally, Figure 6 represents the four ordering nodes (represented by nodes 1–4) used by the proposed solution to add or append new transactions to the network. The numbering from 1–6 in Figure 6, represents the order in which various nodes (*client*, *peers*, and *ordering* nodes) interact with each other to achieve certain objectives within the network. Therefore, the following items explore this information flow in detail:

- 1– the *client* node makes use of the REST-API to connect to the peer nodes within the network. Once the connection is established, the network enables the *client* node to submit a transactional proposal to the network, requesting to either “query” or “update” the ledger.
- 2– the *peer* nodes accept the proposal and examine it using chaincode to verify whether it meets all the requirements associated with it or not. Thereafter, the *peer* node is going to endorse the proposal by attaching a digital signature and generate a proposal response for that request – thus, the “query” process is now complete. The endorsement process determines whether the transaction is accepted or rejected since it reflects the results associated with the pre-defined conditions stipulated within the chaincode.

- 3– the “update” process continues when the client node builds a transaction using the proposal responses submitted by various *peer* nodes and forwards it to the ordering service.
- 4– the ordering service collects all these transactions across the network and groups them into blocks, which are then forwarded to all the peer nodes within the network for validation and committing. The validation process seeks to check the integrity of these transactions using chaincode, while the committing process seeks to append the transaction to the ledger.
- 5– once all the *peer* nodes have updated their ledger, then an event is generated to notify the client node about the completion of the “update” process.
- 6– signifies other actors within the network accessing the updated details appended within the ledger.

All these processes seek to ensure that the ledger within the *peer* nodes is kept up-to-date across the network.

VI. Fictional use case scenario

This section explores a fictional use-case scenario used to expand the idea behind the proposed solution, which includes identifying a problem and providing a counter-solution to it using the ShareTendPro model. Therefore, the following items present the process that takes place within the scenario and these processes are visualised later in Figure 7.

- 1– the LM opens tendering project X for bidding.
- 2– various suppliers apply for tender project X by submitting tender documents to the LM.
- 3– the tendering committee assigned by the LM assesses all the suppliers who applied for project X and submits the results of the assessment to the LM.
- 4– the LM awards project X to Supplier S based on the outcomes presented by the tendering committee.
- 5– the LM assigns Peter to manage project X. Thereafter, Peter uses computer LM_N0 (which stands for LM node 0) to issue a progress report for project X as part of his responsibilities which seeks to portray the following progress “so far, 20% of project X was completed within four months”.

- 6– Peter shared this report with John from the Auditor’s Firm (AF) who was tasked to audit the financial expenditure of tendering project X. Hence, the report acts as proof of payments associated with the work that was completed by Supplier S.
- 7– Peter also shared this report with David from the Investigator’s Firm (IF) who was tasked to investigate allegations of corruption in tendering project X. The report acts as proof of work completed by Supplier S.
- 8 – later on, the DM opens tender project Y for bidding. Assume that project Y is similar to project X.
- 9– assume that Supplier S decided to collude with Peter when it comes to falsifying the report of project X to portray the following progress “50% of project X was completed within four months”.
- 10– various suppliers apply for project Y, including Supplier S. Assume that Supplier S has included a falsified progress report of project X when applying or bidding for project Y and included Peter as a referee who can provide more clarifications regarding project X.
- 11– the DM assigns Martha from the tendering committee of project Y a task to request a progress report of project X from Peter as part of trying to confirm whether Supplier S managed to complete 50% of the project within four months or not. Note that Martha used computer DM_N0 (which stands for DM node 0) to send an electronic mail (email) to Peter when requesting the progress report of project X.
- 12– Peter submitted a falsified progress report of project X to Martha (DM_N0) at the DM.
- 13– the tendering committee of the DM assesses all the suppliers who applied for project Y and submits the results of the assessment to the DM.
- 14– the DM awards tendering project Y to Supplier S based on the outcome of the assessment which was motivated by the information provided by the supplier and confirmed by Peter who works at the LM.

The main objective of this scenario was to depict a loophole that might be used to tamper with the project information in such a way that it can be used to influence the decision of other projects offered by a different municipality. For instance, in the scenario, a falsified report of project X was used to influence the decision when it comes to awarding project Y offered by the DM. Figure 7 seeks to visualise this scenario as various actors in different organisations interact with either a falsified or a legit report of project X. Assume that the communication

mechanism used to share the report of project X was an email. Hence, Figure 7 depicted the computers used by various actors in different organisations as they interact with an electronic report of project X.

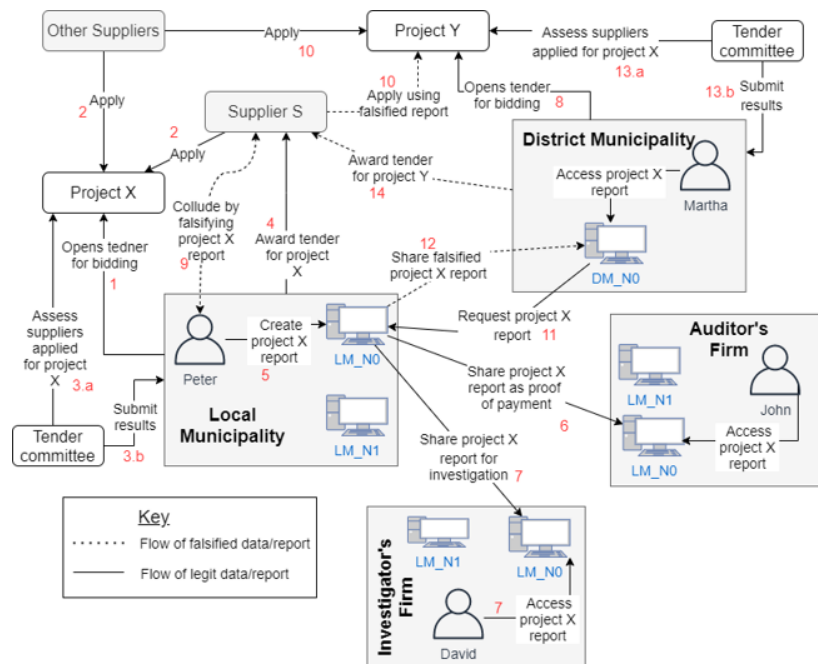


FIGURE 7: SCENARIO PROBLEM

To support the current tendering system, this study proposed a distributed model, instead of conventional email, that seeks to connect all the computers of various organisations that have an interest in the tendering project. For instance, the computers that have an interest in project X are LM_N0 (i.e., LM node 0), DM_N0 (i.e., DM node 0), IF_N0 (i.e., IF node 0), and AF_N0 (i.e., AF node 0) as shown in Figure 7. Therefore, the proposed model would be used as a tool that replaces email when it comes to sharing project information with all the people that have an interest in the tendering project. Additionally, the establishment of the Blockchain network also allows these computers to share project information securely while preserving the integrity of the information. The establishment of the ShareTendPro network as a solution is also aimed at enforcing trust and transparency among various organisations that have an interest in the tendering project.

Figure 8 depicts how this study addresses the identified problem within the scenario by introducing the ShareTendPro network as a solution. A more detailed discussion of the ShareTendPro solution as shown in Figure 8 follows next in an attempt to solve the problem shown in Figure 8.

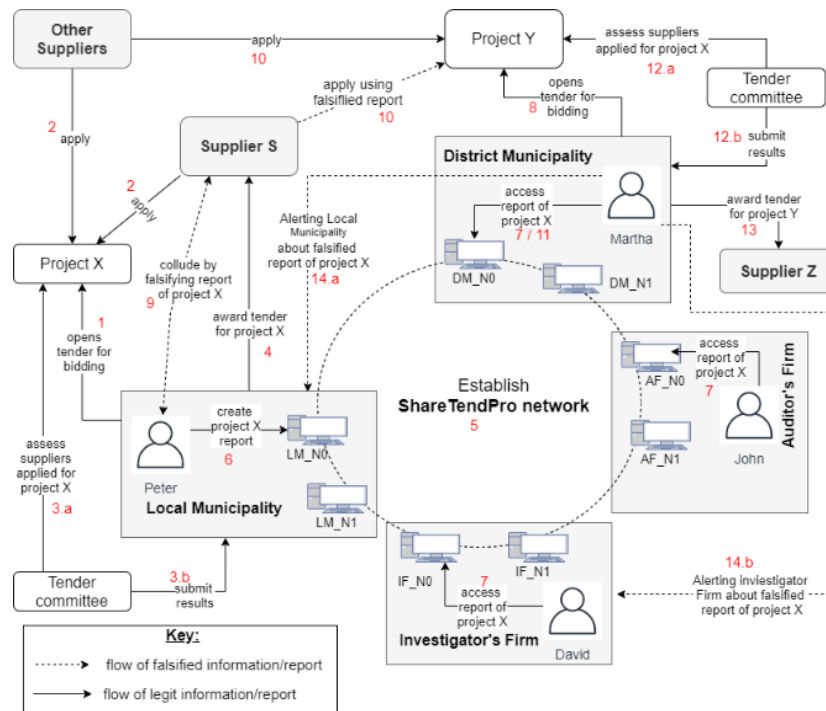


FIGURE 8: SHARETENDPRO SOLUTION

The process taking place within the ShareTendPro solution (which is Figure 8) is as follows:

- 1–4: these steps are similar to steps 1-4 as discussed in the scenario of Figure 7.
- 5– represents the establishment of the ShareTendPro network that would be used to share project information securely while preserving the integrity of the information.
- 6– depicts Peter using computer LM_N0 to create a progress report of project X. Note that computer LM_N0 is one of the computers of the LM that has joined the ShareTendPro network. Hence, the report created by Peter would be stored within the blockchain of the ShareTendPro network.
- 7– depicts various computers accessing the report of project X that was created using computer LM_N0. Note that this step is automatically activated when computer LM_N0 submits the report of project X to the Blockchain in the ShareTendPro network, whereby the ShareTendPro network distributes it to all the computers that have joined the communication channel, due to the inner workings of the HLF framework.
- 8– depicts the DM opening project Y for bidding. This step is similar to step 8 of Figure 7.
- 9– depicts Supplier S and Peter colluding by falsifying the report of project X. This step is similar to step 9 of Figure 7. Later it will become clear how this falsification is detected.

- 10– depicts various suppliers applying for tendering project Y offered by the DM, including Supplier S. This step is similar to step 10 of Figure 7. Assume that Supplier S has included the falsified report on the tendering documents when bidding for project Y.
- 11– represents Martha who was tasked by the tendering committee of project Y to confirm the progress report submitted by Supplier S within the ShareTendPro network. Note that Martha at node DM_N0 did not request the report of project X as compared to the scenario depicted in step 11 of Figure 7 because the report is now available in the ShareTendPro network as she can access it directly.
- 12– depicts the tendering committee of the DM assessing all the suppliers that have applied for project Y and submitting the results of the assessment to the DM. However, the tendering committee realised that the report (i.e. document) of project X submitted by Supplier S contradicts the actual details (i.e. report) stored within the Blockchain of the ShareTendPro network. Due to this discrepancy, Supplier S is removed from the bidding process of project Y with consequences, and another supplier will need to be appointed.
- 13– depicts the DM awarding tender project Y to supplier Z. Note that this was achieved after penalising Supplier S since the information or report provided by the supplier does not correspond with the actual report stored within the ShareTendPro network.
- 14– represents Martha who is part of the tendering committee of project Y alerting the LM and IF about the falsified report of project X for further investigations. The LM will conduct an internal investigation to discipline Peter, while the IF will conduct corruption-related activities or investigations between Peter and Supplier S which include acts of bribery. This, however, is out of the scope of this research and will not be shown further.

This section has outlined the theoretical use-case scenario and its solution with an aim of visualising how the proposed model handles such issues. Therefore, the following section focuses on the results of the ShareTendPro prototype.

VII. ShareTendPro prototype results

There are various results that were generated by the ShareTendPro prototype during the testing phase. Some of these results were associated with the Blockchain network configurations and the operational network testing. However, this section only focuses on the results associated

with operational testing of the proposed solution because it seeks to either create, modify, query, and delete project information within the network. Hence, the following subsections explore the results generated by these operations starting from “*creating tender*”, “*querying tender*”, “*updating tender*”, and “*deleting tender*”, as well as “*querying tender history*”.

A. Creating tender

The results generated by the “*creating tender*” process (which is step 6 of Figure 8) depict the creation of a tendering project report by computer LM_N0 as shown in Figure 9. Therefore, lines 202–212 of Figure 9 depict a function called *chaincodeInvokeByAddingProject()* that contains a command line used by peer0 (i.e. corresponding to LM_N0) of the LM to create the tendering project information of project X. Line 203 of Figure 9 calls a function called *setGlobalsForPeer0LocalMunicipality()* that enables the Blockchain network to initialise peer0 (LM_N0) in the LM. Lines 204–211 represent the command line that creates tendering project information within the Blockchain network. Line 210 represents a flag that seeks to execute a function called *createTender()* contained within the chaincode with the following five arguments namely:

- args[0] = “Tender1000” (1)
- args[1] = “Local Municipality” (2)
- args[2] = “Project X” (3)
- args[3] = “20% of the project has been completed within four months” (4)
- args[4] = “Supplier S” (5)

```

202 chaincodeInvokeByAddingProject() {
203     setGlobalsForPeer0LocalMunicipality
204     peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer1.orderer.org.workplace \
205     --tls SCORE_PEER_TLS_ENABLED --cafile $ORDERER_CA -C $CHANNEL_NAME -n ${CC_NAME} \
206     --peerAddresses localhost:7051 --tlsRootCertFiles $PEER0_LocalMun_CA \
207     --peerAddresses localhost:9051 --tlsRootCertFiles $PEER0_DistrictMun_CA \
208     --peerAddresses localhost:11051 --tlsRootCertFiles $PEER0_InvestigatorFir_CA \
209     --peerAddresses localhost:13051 --tlsRootCertFiles $PEER0_AuditorFir_CA \
210     -c '{"function": "createTender", "Args": ["Tender1000", "Local Municipality", "Project X",
211     "20% of the project has been completed within four months", "Supplier S"]}'
212 }
213 chaincodeInvokeByAddingProject

```

The terminal window shows the following output:

```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
2021-09-14 20:02:53.120 SAST [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200 payload:{"tenderOwner":"Local Municipality","\projectName":"Project X","\reports":"20% of the project has been completed within four months","\supplier":"Supplier S"}
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$

```

FIGURE 9: CREATE TENDER

Note that this project report is identified using the key (1). Label R of Figure 9 represents the results displayed within the command prompt after executing the *chaincodeInvokeByAddingProject()* function, as presented by line 213. Additionally, the command line returns a successful status of 200 which implies that the information of the

tendering project X was successfully created within the Blockchain network, as shown in label R.

After the project information has been added to the Blockchain network, that information would automatically be available for other nodes to access it. The following section focuses on the results generated by the “*query tender*” process. Note that other nodes use the key (1) to access the project report created in Figure 9.

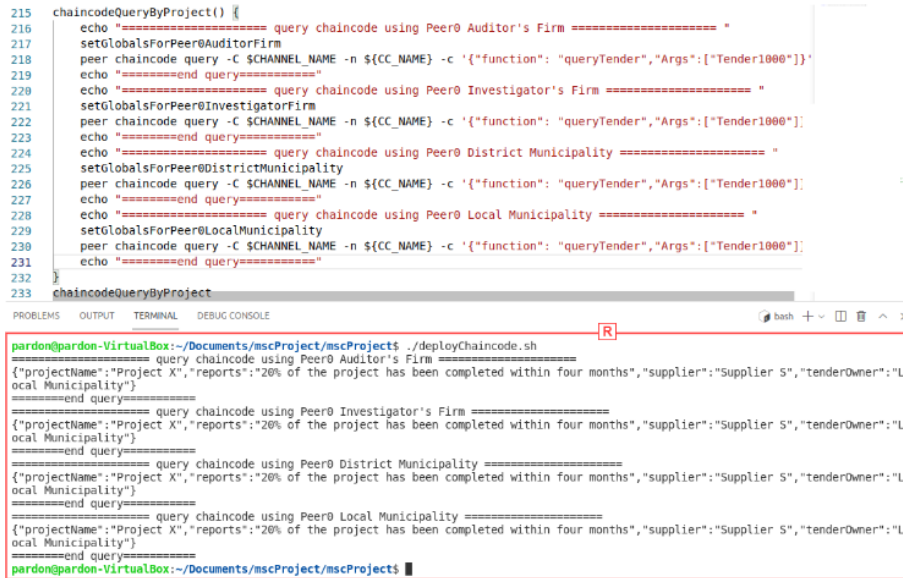
B. Querying tender

The results generated by the “*querying tender*” process (which is step 7 of Figure 8) depict an example of how nodes can access project X information using the key (1), as shown in Figure 10. Therefore, lines 215–232 of Figure 10 depict the details of a function called *chaincodeQueryByProject()*, which contains the command lines used by various nodes to access the report of tendering project X. For instance, lines 217 and 218 represent the command used by peer0 (i.e. AF_N0) of the AF to access project X information (using (1)). Additionally, line 2017 seeks to call a function called *setGlobalsForPeer0AuditorFirm()* that enables the Blockchain network to initialise AF_N0, while line 218 depicts the command line used by AF_N0 to access project X information using the key (1). Note that the command line makes use of a function called *queryTender()* contained within the chaincode to interact with the Blockchain network, as shown in Figure 10. The results of this command line are displayed within the command prompt represented by label R. However, the same notion applied to the command line represented in line 218 can also be used to other command lines used by other nodes. Label R of Figure 10 represents the results generated after executing the *chaincodeQueryByProject()* function using line 233. Hence, the results contained within this process also portray the distributed nature of the ShareTendPro model because other nodes automatically have access to the report of tendering project X compared to the scenario depicted in Figure 7.

```

215 chaincodeQueryByProject() {
216     echo "===== query chaincode using Peer0 Auditor's Firm ===== "
217     setGlobalsForPeer0AuditorFirm
218     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} -c '{"function": "queryTender", "Args":["Tender1000"]}'
219     echo "=====end query===== "
220     echo "===== query chaincode using Peer0 Investigator's Firm ===== "
221     setGlobalsForPeer0InvestigatorFirm
222     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} -c '{"function": "queryTender", "Args":["Tender1000"]}'
223     echo "=====end query===== "
224     echo "===== query chaincode using Peer0 District Municipality ===== "
225     setGlobalsForPeer0DistrictMunicipality
226     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} -c '{"function": "queryTender", "Args":["Tender1000"]}'
227     echo "=====end query===== "
228     echo "===== query chaincode using Peer0 Local Municipality ===== "
229     setGlobalsForPeer0LocalMunicipality
230     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} -c '{"function": "queryTender", "Args":["Tender1000"]}'
231     echo "=====end query===== "
232 }
233 chaincodeQueryByProject

```



```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProjects$ ./deployChaincode.sh
===== query chaincode using Peer0 Auditor's Firm =====
{"projectName":"Project X","reports":"20% of the project has been completed within four months","supplier":"Supplier S","tenderOwner":"Local Municipality"}
=====end query=====
===== query chaincode using Peer0 Investigator's Firm =====
{"projectName":"Project X","reports":"20% of the project has been completed within four months","supplier":"Supplier S","tenderOwner":"Local Municipality"}
=====end query=====
===== query chaincode using Peer0 District Municipality =====
{"projectName":"Project X","reports":"20% of the project has been completed within four months","supplier":"Supplier S","tenderOwner":"Local Municipality"}
=====end query=====
===== query chaincode using Peer0 Local Municipality =====
{"projectName":"Project X","reports":"20% of the project has been completed within four months","supplier":"Supplier S","tenderOwner":"Local Municipality"}
=====end query=====
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProjects$

```

FIGURE 10: QUERY TENDER

After the project information of tendering project X is distributed to other nodes, we can then proceed with a process that seeks to update some of the details contained within the project report. The following section focuses on the results generated by a process that seeks to update the details of the supplier and project report. Note that the update process might still be used for illegal altering of project information by some of these authorised nodes. However, the project history that would be discussed later can be used as a mechanism that seeks to explore what has transpired within that project since the information stored within the Blockchain data of the ledger is immutable by default as indicated earlier on.

C. Updating tender

The results generated by the “*updating tender*” process depict a process that seeks to modify some of the information of a particular tendering project (i.e., project X in this case). However, this section explores the two possible updating processes implemented within the ShareTendPro network which are “*update supplier details*” and “*update tender report details*”. The “*update supplier details*” process focuses on the results generated during a process that seeks to update the details of the supplier (e.g., contact details or to assign a new supplier altogether). The “*update tender report details*” process focuses on the results generated by a process that seeks to update the report of the tendering project (e.g., doing editorial updates).

1. Update supplier details

Figure 11 depicts the results generated by a process that seeks to update supplier information on tendering project X. For instance, lines 235–244 of Figure 11 depict a function called *InvokeByChangingSupplier()*, which is a function used by peer0 (i.e., computer LM_N0) of the

LM to update the details of a supplier from “*Supplier S*” to “*Supplier Z*”. As indicated in lines 237–243, the command makes use of a function called *changeTenderSupplier()*, as seen in line 243, contained within the chaincode to interact with the Blockchain network. The *changeTenderSupplier()* function requires two arguments. The first argument represents a key (1) used to identify a tendering project, while the second argument is used to assign the updated details of the supplier which is “*Supplier Z*” in this case. Therefore, the results displayed (as presented by label R) are generated after executing the *InvokeByChangingSupplier()* function using line 245. The status of the results reflected within label R is 200, which implies that the supplier details were updated successfully.

```

235 InvokeByChangingSupplier() {
236     setGlobalsForPeer@LocalMunicipality
237     peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer1.orderer0.org.workplace \
238     --tls $CORE_PEER_TLS_ENABLED --cafile $ORDERER_CA -C $CHANNEL_NAME -n ${CC_NAME} \
239     --peerAddresses localhost:7051 --tlsRootCertFiles $PEER0_LocalMun_CA \
240     --peerAddresses localhost:9051 --tlsRootCertFiles $PEER0_DistrictMun_CA \
241     --peerAddresses localhost:11051 --tlsRootCertFiles $PEER0_InvestigatorFir_CA \
242     --peerAddresses localhost:13051 --tlsRootCertFiles $PEER0_AuditorFir_CA \
243     -c '{"function": "changeTenderSupplier", "Args": ["Tender1000", "Supplier Z"]}'
244 }
245 InvokeByChangingSupplier

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

R

```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$ ./deployChaincode.sh
2021-09-14 20:12:19.464 SAST [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 001 Chaincode invoke successful. result: status:200
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProject$

```

FIGURE 11: UPDATE TENDER SUPPLIER

After updating the details of the supplier, the second process that seeks to update the details of a project report is considered. Hence, the following section explores the results generated by a process that seeks to update the project report of a tendering project.

2. Update tender report details

Note that the results generated using the “*update tender report details*” process are similar to the results generated by the “*update supplier details*” process (as depicted in Figure 11). Hence, the in-depth details of the results generated by the “*update tender report details*” process are not explored in detail to avoid the repetition of some of the concepts. Furthermore, note that the “*update tender report details*” process seeks to modify the project report to reflect the new report that portrays the following progress “*50% of the project was completed within four months*”.

To verify whether indeed the project information (which is both supplier and report details) was updated successfully within the ShareTendPro network, then we can use a “*query tender*” process to access the latest updates of the tendering project X using the key (1). Therefore, Figure 12 depicts the results generated by the “*query tender*” process. Note that the project information was successfully updated since it has changed:

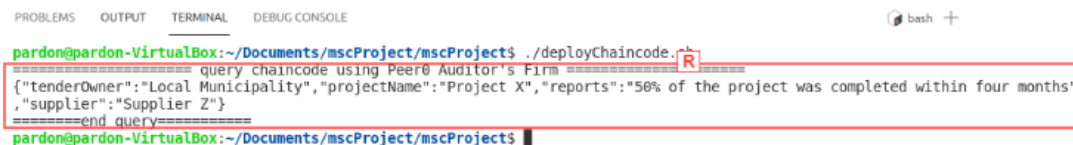
- **From:** “20% of the project was completed within four months”, as shown in Figure 10.

- **To:** “50% of the project was completed within four months”, as depicted in Figure 12.
- **From:** “Supplier S”, as presented in Figure 10.
- **To:** “Supplier Z”, as shown in Figure 12.

```

267 chaincodeQueryByProject() {
268     echo "===== query chaincode using Peer0 Auditor's Firm ====="
269     setGlobalsForPeer0DistrictMunicipality
270     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} \
271         -c '{"function": "queryTender","Args":["Tender1000"]}'
272     echo "=====end query===== "
273 }
274 chaincodeQueryByProject

```



```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProjects$ ./deployChaincode.R
===== query chaincode using Peer0 Auditor's Firm =====
{"tenderOwner":"Local Municipality","projectName":"Project X","reports":"50% of the project was completed within four months",
"supplier":"Supplier Z"}
=====end query=====
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProjects$

```

FIGURE 12: QUERY UPDATED TENDER

After updating the details of a tendering project, one might decide to delete the project information stored within the ShareTendPro network. Assume that this process was activated by a withdrawal of the tendering project by a supplier that results in the termination of a contract between “Supplier Z” and the LM. Therefore, the following section explores the results generated by a process that seeks to delete project information stored within the ShareTendPro network. As indicated in the previous section, the evidence of the existence of tendering project X will not be completely lost because this information is still stored in the Blockchain data of the ledger and it is immutable by default (which implies that it cannot be changed or permanently deleted for historic investigation purposes).

D. Deleting tender

The concept used to implement the “*deleting tender*” process within the ShareTendPro network is similar to the concept used to implement the update tender supplier process (depicted in Figure 11). Hence, the details of the *delete tender* process are not explored in detail to avoid the repetition of some of the concepts. However, the main difference is that the *delete tender* process calls a *deleteTender()* function contained within the chaincode instead of calling *updateTenderSupplier()* function. The *deleteTender()* function only takes one argument which is the key (1).

After performing all the processes that seek to either create, update or delete the project information of a particular tendering project, then the evidence related to such processes is stored within the Blockchain data. As indicated earlier on, the information stored within the Blockchain data is immutable since it seeks to preserve the evidence of what transpired within that specific tendering project. Additionally, the evidence contained within the Blockchain data

can also be used as forensic data since it portrays the entire history of a particular tendering project. Therefore, the following section focuses on the results generated by a process that seeks to access a project history of tendering project X.

E. Querying tender history

The results generated by the “*querying tender history*” process depict how various nodes can access the entire project history of a particular tendering project, which is project X in this case. Therefore, lines 296–302 of Figure 13 depict a function called *chaincodeQueryProjectHistory()*, which contains the details used by a command line that seeks to access the project history of a tendering project associated with the key (1). As indicated in lines 299 and 300, the command makes use of a function called *getHistoryForTender()*, as seen in line 300, contained within the chaincode to interact with the Blockchain network. The results displayed (as presented by label R) are generated after executing the *chaincodeQueryProjectHistory()* function using line 303. Note that the project information displayed within label R contains various transaction IDs and each of these transaction IDs represents a specific process that was executed using the key (1). Furthermore, note that these processes have been executed already, as follows. The processes that were executed using key (1) are creating a tender project (as shown in Figure 9), updating the tender supplier (as shown in Figure 11), updating the tender report, and deleting a tender project. All these processes are identified using a unique transaction ID (represented by TxID in label R). Again, note that the status of a tag “IsDelete” is only true when it comes to the Transaction ID that represents a process that seeks to delete the information of a tendering project, else it is false.

```

296 chaincodeQueryProjectHistory() {
297     echo "===== query chaincode using Peer0 Auditor's Firm ====="
298     setGlobalsForPeer0DistrictMunicipality
299     peer chaincode query -C $CHANNEL_NAME -n ${CC_NAME} \
300     -c '{"function": "getHistoryForTender", "Args": ["Tender1000"]}'
301     echo "=====end query===== "
302 }
303 chaincodeQueryProjectHistory

```

```

pardon@pardon-VirtualBox:~/Documents/mscProject/mscProjects$ ./deployChaincode.sh
===== query chaincode using Peer0 Auditor's Firm =====
[{"TxId": "a35fa2ec5f2225a5befc9db3936facb3cac0c194d3d3ac215a67c2f251616be68", "Value": null, "Timestamp": "2021-09-15 12:28:09.409964752 +0000 UTC", "IsDelete": "true"}, {"TxId": "919896445f3550fbeb11c3011140090b5219d9593b9fd7643dac91ed6d31030f", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "50% of the project was completed within four months", "supplier": "Supplier Z"}, "Timestamp": "2021-09-15 12:24:46.098319371 +0000 UTC", "IsDelete": "false"}, {"TxId": "b31185f27ae9cd52119e1e81d7fdbdc2b8e369281e15f227748c6781681e9e9c", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier Z"}, "Timestamp": "2021-09-15 12:28:23.951643768 +0000 UTC", "IsDelete": "false"}, {"TxId": "94a208cc045ee21011580be40f2241e528381eeb59f78bc505a1a75517da258c", "Value": {"tenderOwner": "Local Municipality", "projectName": "Project X", "reports": "20% of the project has been completed within four months", "supplier": "Supplier S"}, "Timestamp": "2021-09-15 12:21:58.628973647 +0000 UTC", "IsDelete": "false"}]
=====end query=====
pardon@pardon-VirtualBox:~/Documents/mscProject/mscProjects$

```

Transaction ID for deleteTender

Transaction ID for updateTenderSupplier

Transaction ID for updateTenderReport

Transaction ID for createTender

FIGURE 13: QUERY TENDER HISTORY

VIII. Evaluation of the research study

The details contained within this section are classified into two subsections namely: the “*benefits of this research study*” and the “*shortcoming of this research study*”. Therefore, the following items provide the details of these two subsections.

A. The benefits of this research study:

1. *Distributed nature of the proposed solution*– it is derived from the use of DL to share project information with all the actors that have an interest in it. This ensures that the participants would not be able to collude with each other over project data since they would have access to the same data.
2. *Enhanced information security*– refers to the mechanisms used to increase the difficulty for unauthorised parties (i.e., hackers) to either access or tamper with project information within the ShareTendPro network. The mechanisms used by the ShareTendPro network to secure project information are: distributing project data in multiple locations, the use of cryptography and a secure communication channel to encrypt and secure project data, and the use of timestamped and immutable transactions for data integrity.
3. *Greater transparency over project information*– is derived from the use of DL because it ensures that the business process used to either create, modify, or delete project information within the ShareTendPro network is transparent. Hence, the authorised actors will not have to worry about accessing project data that might be processed unnoticed. This mechanism disables the rogue parties the ability to collude over project information.
4. *Automated transactions*– refer to the use of chaincode components to reduce human interactions within the ShareTendPro network while increasing the efficiency and speed at which it processes transactions. As indicated in the previous section, all the transactions submitted by various nodes are sent to a specific method contained within the chaincode. Thereafter, the processes that follow next are triggered automatically since the chaincode uses the submitted transactions to perform certain tasks within the network. For instance, computer LM_N0 in the previous section submitted a transaction to create a tendering project to a method called *createTender()* contained within the chaincode. Thereafter, the chaincode automatically executed a process that seeks to create a tendering project within the Blockchain network and return the results to the computer that submitted the transaction.

When the process of creating a tendering project is completed, another process is initiated automatically, which is to distribute the project information that was created to other nodes that form part of the network. Hence, the transaction that seeks to create tendering project information within the ShareTendPro network consists of two automated processes, the first process is automatically executed by the chaincode component and the second process is automatically executed by a DL component.

5. *Time efficiency*– refers to the time it takes for a participant to have full access to the entire project history of a specific tendering project of their interest using the ShareTendPro network. Note that the authorised parties within the network have full access to the data collected by the Blockchain component of the ledger since it contains records of the transactions that seek to create, update, or delete project information of a specific tendering project. Therefore, this data is readily available to all the authorised parties who have an interest in knowing what transpired within that tendering project. For instance, having access to such data by the AF enables them to audit the tendering project in real-time, which reduces the time taken to look at the financial documents that portray the expenditure incurred during the execution of the tendering project at the end of a fiscal year or closing period of the desired project. Furthermore, having access to such data by the IF enables them to conduct a real-time investigation without going to the municipality to collect such information in person. Additionally, having access to such data by both the AF and IF enables them to quickly resolve issues related to irregular expenditure or corruption allegations while promoting accountability on other hand.
6. *Credible evidence*– refers to information that can be presented to the court of law as evidence of what has transpired within a specific tendering project. This evidence is collected by an investigator from the ShareTendPro network by accessing the project history of a particular tendering project because the data stored within the Blockchain component of the ledger is immutable, timestamped, and cryptographically encrypted. All these mechanisms of having data that is immutable, time-stamped, and cryptographically encrypted seek to preserve the integrity of the evidence that portrays what transpired within that project.
7. *Promotes real-time auditing and investigations*– refers to information that can be accessed in real-time that allows both auditors or investigators to either audit or investigate a tendering project in real-time. The benefit of having real-time auditing or investigation is derived from the distributed nature of the ShareTendPro network because it enables both auditors and

investigators to access project information in real time. Having access to real-time data allows the auditors or investigators to complete their auditing or investigation process quickly, which also contributes towards having greater accountability over the tendering project, while aiming at saving the municipality's funds that might be spent through irregular expenditure or corruption activities.

8. *Improve collaboration*– refers to the use of the ShareTendPro network to share project information among various actors working on the same tendering project. For instance, the improved collaboration can benefit both the LM and DM when it comes to executing a joint project because the use of the ShareTendPro network will eliminate issues related to collusion over project information. The responsibility of the LM within the joint project might be executing the tendering project, while the responsibility of the DM might be to oversee the project executed by the LM. Hence, the use of a ShareTendPro network by these municipalities will ensure that accountability is achieved because the process of reporting that tendering project would be more transparent compared to the conventional method of sharing project information in a joint project.

B. The shortcoming of this research study:

1. *Lack of political will*– this can only be regarded as a shortcoming because most of these higher-ranking positions or members of the organisations are easily influenced by the political space. Hence, the political will might be reluctant to adopt a solution that seeks to manage issues associated with monitoring the tendering projects since some of them might be linked to these projects.
2. *Training or workshop-related issues*– this study foresees this as a shortcoming because some of the participants might be reluctant to undergo such training, especially when they are already using several systems or applications.

IX. Conclusion and future work

This section provides a summary of this research study while aiming to shed light on the contribution made by this research. However, the details contained within this section are classified into three subsections: a “*recap of the problem statement*”, “*future work*”, and “*concluding remarks*”.

A. Recap of the problem statement

The primary problem this study sought to address is the use of paperwork to share project information among various organisations since it might contribute to illicit altering of project information during the process. This might also affect the fairness, transparency, data integrity, and competitiveness of the tendering system used by the SALG. To address this problem, this study outlined the following research questions:

- **RQ1:** How does the tendering system work in the South African context? This question was answered in the following way. It was addressed by Section III because it explored the background details on how the current tendering system used by SALG work.
- **RQ2:** Is DLT a possible solution to the identified problem? This question was answered in the following way. It was addressed by implementing a Blockchain prototype that might be used to securely share project information among participants that have an interest in it. The ShareTendPro prototype was designed in Section V and the test results of it were discussed in Section VII.
- **RQ3:** How does transparency, accountability, and integrity of data or information in a potential solution work, and how will it contribute to digital forensics? This question was answered in the following way. The transparency, accountability, and data integrity part of this research question was addressed in Sections IV & V. The other information security mechanisms were also highlighted in other sections such as Sections VII & VIII, including the issue that is related to how does the proposed solution contribute to the digital forensic investigation.

B. Future work

The implementation of the ShareTendPro solution shows the potential of sharing tendering project information within the SALG. One of the main benefits is that the proposed solution seeks to enforce collaboration among various organisations that have an interest in tendering project information by providing real-time data. The access to real-time data also assists other organisations such as AF and IF to audit or investigate a specific tendering system in real-time, which also promotes accountability within that tendering project. However, during the research process, some new areas of interest that fall outside the scope of this study were discovered which can be considered for further research. Hence, the following items explore these areas of interest.

- The ShareTendPro network is based on the backend development, therefore, future research can focus on the implementation of a frontend application that would be used as the user interface of the ShareTendPro network.
- To expand the scope of the ShareTendPro network by including other stakeholders that fall within the South African Government, which includes all three spheres: Local, Provincial, and National Government.
- To implement a mechanism that can be used by various computers to temporarily store project information when it becomes inactive and share that information with other nodes within the network when it becomes active.
- To implement a mechanism that can be used to integrate the ShareTendPro model with the existing tendering system with an aim of trying to avoid the reluctance from other participants to adopt the proposed solution.

The following section focuses on the final concluding remarks of the research study.

C. Concluding remarks

The ShareTendPro network demonstrates how BCT as a tool can be used to securely share project information with all parties that have an interest in the tendering project. The ShareTendPro network secures its project information using various information security mechanisms such as DL, cryptographic encryptions, timestamps, and immutable data or transactions. Note that the ShareTendPro network satisfies the four features or issues highlighted in Table 2 because it seeks to monitor the execution of the tendering projects (i.e., project X), rather than focusing on the processes that fall within the e-procurement (i.e., tender bidding process). Additionally, the ShareTendPro network supports a private Blockchain network configuration that does not rely on either gas or mining algorithms to add new transactions to the network.

The SALG uses a tendering system to promote public and private partnerships, therefore, the ShareTendPro network becomes an essential platform for securely sharing project information without colluding. Additionally, the ShareTendPro network will provide greater transparency and accountability over project information, while enforcing trust on the other hand because no one will have to worry about the illegal altering of project information that might be processed unnoticed. This ensures that the ShareTendPro network stores credible digital evidence of the tendering project that can be used to depict the entire project history of a particular project of interest. Furthermore, the ShareTendPro network ensures that all the participants have instant

access to real-time data stored within the Blockchain network without having to worry about issues that emanate from requesting that data directly from a specific organisation because some of them might be reluctant to share it.

The ShareTendPro network, therefore, would provide a revolutionary step towards curbing corruption in countries, like South Africa, where corruption currently enjoys high tide. Additionally, it is hoped that other countries that face similar issues might also adopt this scheme since it can be configured to accommodate various tendering systems used by different countries.

Acknowledgment

This research study was funded and supported by the Council for Scientific and Industrial Research (CSIR) and the University of Pretoria (UP), South Africa. Special thanks go to Prof H. Venter (UP) and Mr. H. Le Roux (CSIR) for their continuous support and contribution towards the success of this research.

Declarations

Conflict of Interest: The authors declare that they have no conflict of interest.

Ethical approval: This article does not contain any data associated with human or animals participants performed by any of the authors.

Availability of data, materials or code: The code is available upon request.

References

- [1] N. Kshetri, "Will Blockchain emerge as a tool to break the poverty chain in the Global South?," *Third World Quarterly*, vol. 38, no. 8, pp. 1710-1732, 2017.
- [2] Z. Arsovski, P. Lula and A. Dordevic, "IMPACT OF ICT ON QUALITY OF LIFE," in *Center for Quality*, 2016.
- [3] Department of Science and Technology (RSA), "ICT RDI Roadmap," [Online]. Available: <http://digitaladvantage.co.za/the-ict-rdi-roadmap/>. [Accessed 30 August 2018].
- [4] S. Ngobeni, "An analysis of the tender process in national government in South Africa," in *MBA Thesis, North-West University, Potchefstroom Campus*, Potchefstroom, 2011.
- [5] A. Carstensen and J. Bernhard, "Design science research—a powerful tool for improving methods in engineering education research," *European Journal of Engineering Education*, vol. 44, no. 1-2, pp. 85-102, 2019.
- [6] ASTRI, "Whitepaper on Distributed Ledger Technology," 11 November 2016. [Online]. Available: <https://www.astri.org/tdprojects/whitepaper-on-distributed-ledger-technology/>. [Accessed 18 02 2019].
- [7] H. Natarajan , S. Krause and H. Gradstein , "Distributed Ledger Technology and Blockchain," The World Bank Group: Open knowledge Repository, 2017.

- [8] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," www.bitcoin.org, 2008.
- [9] W. Dai, "B-money," 1998. [Online]. Available: <http://www.weidai.com/bmoney.txt>. [Accessed 28 01 2019].
- [10] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84-90, 1981.
- [11] K. Wüst and A. Gervais, "Do you need a Blockchain?," in *In 2018 Crypto Valley Conference on Blockchain Technology (CVCBT) (pp. 45-54)*, 2018.
- [12] bitcoin, "Mining," bitcoin wiki, [Online]. Available: <https://en.bitcoin.it/wiki/Mining>. [Accessed 30 05 2019].
- [13] R. Zagone, "Results of the Bank of England/Ripple Proof of Concept Published Today," Ripple | Insight, 10 July 2017. [Online]. Available: <https://ripple.com/insights/results-of-the-bank-of-englandripple-proof-of-concept-published-today/>. [Accessed 27 03 2019].
- [14] Ripple, "XRP the digital asset for payments," Ripple | XRP, [Online]. Available: <https://ripple.com/xrp/>. [Accessed 28 03 2019].
- [15] <https://xrpl.org>, "Docs: Introduction to Consensus," XRP Ledger, [Online]. Available: <https://xrpl.org/intro-to-consensus.htmlxrpl.prg>. [Accessed 25 11 2019].
- [16] D. Schwartz, N. Youngs and A. Britto, "The ripple protocol consensus algorithm," Ripple Labs Inc White Paper 5(8), 2014.
- [17] Ethereum, "What is Ethereum?," Ethereum.org, [Online]. Available: <https://ethereum.org/en/what-is-ethereum/>. [Accessed 02 11 2021].
- [18] R. Shah and S. Rajagopal, "M-DPS: a blockchain-based efficient and cost-effective architecture for medical applications," *International Journal of Information Technology*, vol. 14, no. 4, pp. 1909-1921, 2022.
- [19] P. Hegedus, "Towards Analyzing the Complexity Landscape of Solidity Based Ethereum Smart Contracts," in *2018 IEEE/ACM 1st International Workshop on Emerging Trends in Software Engineering for Blockchain (WETSEB)*, Gothenburg, Sweden, Sweden, 2018.
- [20] Z. Wang, H. Jin, W. Dai, K. Choo and D. Zou, "Ethereum smart contract security research: survey and future research opportunities," *Frontiers of Computer Science*, vol. 15, no. 2, pp. 1-18, 2021.
- [21] S. Batchu, O. S. Henry and A. Hakim, "A novel decentralized model for storing and sharing neuroimaging data using ethereum blockchain and the interplanetary file system," *International Journal of Information Technology*, vol. 13, no. 6, pp. 2145-2151, 2021.
- [22] A. Joshi, M. Han and Y. Wang, "A survey on security and privacy issues of blockchain technology," *Mathematical Foundations of Computing*, vol. 1, no. 2, pp. 121-147, 2018.
- [23] Hyperledger, "About," Hyperledger Projects, [Online]. Available: <https://www.hyperledger.org/about>. [Accessed 19 03 2019].
- [24] A. Baliga, "Understanding Blockchain consensus models," In Presistent, 2017.

- [25] M. Vukolic, "Hyperledger Fabric: An open-source distributed operating system for permissioned blockchains," 22 June 2017. [Online]. Available: <https://archiveweb.epfl.ch/blockchain-summer.epfl.ch/talks/hyperledger-fabric-vukolic.pdf>. [Accessed 29 03 2019].
- [26] M. Vukolić, "Hyperledger fabric: towards scalable blockchain for business," IBM Research, 17 June 2016. [Online]. Available: https://trustindigitallife.eu/wp-content/uploads/2016/07/marko_vukolic.pdf. [Accessed 29 03 2019].
- [27] P. Sajana, M. Sindhu and M. Sethumadhavan, "On blockchain applications: hyperledger fabric and ethereum," *International Journal of Pure and Applied Mathematics*, vol. 118, no. 18, pp. 2965-2970, 2018.
- [28] S. Rouhani and R. Deters, "Performance analysis of ethereum transactions in private blockchain," in *In 2017 8th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2017.
- [29] C. Cachin, "Architecture of the hyperledger blockchain fabric," in *In Workshop on distributed cryptocurrencies and consensus ledgers*, 2016.
- [30] Hyperledger-fabric, "Key concepts: Ledger," Hyperledger-fabric documents release-1.4, [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-1.4/ledger/ledger.html>. [Accessed 09 October 2019].
- [31] D. Mali, D. Mogaveera, P. Kitawat and M. Jawwad, "Blockchain-based e-tendering system," in *2020 4th International Conference on Intelligent Computing and Control Systems*, pp. 357-362, 2020.
- [32] X. Li, "BCES: A BlockChain based Credible E-Bidding System," in *IEEE 6th International Conference on Computer and Communications*, 2020.
- [33] T. Weingärtner, D. Batista, S. Köchli and G. Voutat, "Prototyping a Smart Contract Based Public Procurement to Fight Corruption.," *Computers*, vol. 10, no. 7, p. 85, 2021.
- [34] Y. Goswami, A. Agrawal and A. Bhatia, "E-Governance: A Tendering Framework Using Blockchain with Active Participation of Citizens," in *2020 IEEE International Conference on Advanced Networks and Telecommunications Systems*, 12 2020.
- [35] Linux Foundation Projects, "Hyperledger-composer," Hyperledger Foundation, August 2021. [Online]. Available: <https://www.hyperledger.org/use/composer>. [Accessed 16 05 2022].
- [36] J. Deshpande, M. Gowda, M. Dixit, M. Khubbar, B. Jayasri and S. Lokesh, "Permissioned blockchain based public procurement system," *Journal of Physics: Conference Series*, vol. 1706, no. 1, 2020.
- [37] I. Omar, R. Jayaraman, M. Debe, K. Salah, I. Yaqoob and M. Omar, "Automating procurement contracts in the healthcare supply chain using blockchain smart contracts," in *EEE Access*, 9, 2021.
- [38] D. Čeke, N. Buzadija and S. Kunosić, "Enhancing transparency and fairness in public procurement process with the support of blockchain technology: a smart contract based approach," *21st International Symposium INFOTEH-JAHORINA*, March 2022.
- [39] F. Zbinden and G. Kondova, "Economic development in Mexico and the role of blockchain," *Advances in Economics and Business*, vol. 7, no. 1, pp. 55-64, 2019.

- [40] M. Das, X. Tao, Y. Liu and J. Cheng, "A blockchain-based integrated document management framework for construction applications," in *Automation in Construction*, 133, 104001, 2022.
- [41] V. Hassija, V. Chamola, D. Krishna, N. Kumar and M. Guizani, "A blockchain and edge-computing-based secure framework for government tender allocation," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2409-2418, 2020.
- [42] S. Perera, S. Nanayakkara, M. Rodrigo, S. Senaratne and R. Weinand, "Blockchain technology: Is it hype or real in the construction industry?," *Journal of Industrial Information Integration*, vol. 17, p. 100125, 2020.
- [43] F. Hardwick, R. Akram and K. Markantonakis, "Fair and Transparent Blockchain Based Tendering Framework - A Step Towards Open Governance," *17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications*, 2018.
- [44] O. Ogunlela, O. Ojugbele and R. Tengeh, "Blockchain technology as a panacea for procurement corruption in digital era," *International Journal of Research in Business and Social Science*, vol. 10, no. 4, 2021.
- [45] S. Quamara and A. Singh, "SChain: towards the quest for redesigning supply-chain by augmenting Blockchain for end-to-end management," *International Journal of Information Technology*, pp. 1-12, 2022.
- [46] Hyperledger Composer, "Introduction," [Online]. Available: <https://hyperledger.github.io/composer/latest/introduction/introduction.html>. [Accessed 18 12 2019].
- [47] S. Balamurugan, A. Ayyasamy and K. S. Joseph, "IoT-Blockchain driven traceability techniques for improved safety measures in food supply chain," *International Journal of Information Technology*, vol. 14, no. 2, pp. 1087-1098, 2022.
- [48] Docker, "What is a Container?," Docker, [Online]. Available: <https://www.docker.com/resources/what-container>. [Accessed 09 09 2020].
- [49] P. Ramazhamba, T. Mashiane and Z. Dlamini, "Grade monitoring system: A prototype for Thulamela secondary schools," in *IST-Africa Week Conference (IST-Africa)*, 2018.

D.4. ShareCrime: A Blockchain concept for sharing criminal information in Criminal Justice Systems (submitted to the IFIP WG 11.9 International Conference on Digital Forensics)

Abstract

Criminal Justice Systems across the world, specifically in the South African context, still faces various issues that can be associated with missing case dockets or digital evidence. Another notable issue is the mechanisms used to share criminal data as some of the processes still rely on email or paper-based document exchange, which might contribute to the illegal altering of criminal data for corruption purposes. Therefore, this study proposes a blockchain concept that might be used to share criminal data securely and efficiently with all the law enforcement entities that have an interest in it. A use-case diagram was used to depict the implementation of the proposed model. It is recommended that the adoption of the proposed model will enable various parties to access criminal data in real-time, which might contribute to the early delivery of justice to the affected parties. Moreover, the proposed model also seeks to improve collaboration among various parties, especially when it comes to joint operations or investigations between the South African Police Service (SAPS) and the National Prosecuting Authority (NPA). The proposed model also stores credible evidence since its data is immutable, which implies that it cannot be deleted.

Keywords—Blockchain technology, criminal justice systems, digital evidence sharing.

I. Introduction

We are living in a digital era whereby most of the people within our society interact with digital information in one way or the other. Access to this digital information is driven by the advancement of information communication technologies (ICTs) we use in our day-to-day activities. One of the main advantages that drive the adoption of ICTs is the way it enables us to collect, store, process, and analyse digital information. As we interact with this digital information, we tend to leave a digital footprint of what happened, when, and where. This digital footprint can be viewed as digital evidence. However, when a crime is committed, the report of a forensic investigator must seek to answer the following questions: “What happened?”, “Where did it happen?”, “When did it happen?”, “Who might be involved?”, and suggest “Why it happened?”, including trying to explain “How it happened?”. All these questions (what, where, when, who, why, and how) are known as the 5 Ws and H questions. All these questions play

critical roles within the criminal justice process because they seek to prove that a particular individual is linked to a specific criminal activity. Therefore, to preserve such crucial criminal information that would be used to either convict criminal syndicates or release the accused individual, requires the adoption of well-secured and efficient innovative ICT solutions, especially when it comes to sharing such information with interested parties without colluding amongst each other.

For instance, the South African Police Service (SAPS) has introduced the use of the Integrated Case & Docket Management System (ICDMS) after experiencing an increase in the trend of missing paper-based case dockets. The main aim of introducing the ICDMS was to combat issues associated with lost or stolen case dockets or evidence because, in the 2009 parliamentary questioning, the SAPS revealed that 688 dockets went missing between April 2008 and February 2009 [1]. Additionally, in 2020, the findings of the Department of Community Safety's Docket Archive Store Assessment Project in the Western Cape reported that approximately 63% of the case dockets were lost on the archive system without any disciplinary action, while 14% of them were lost in court [2][3]. Carte Blanche [4] has also reported how case dockets containing criminal data are being sold by a SAPS official who works under the administration division before they can be captured into the ICDMS. Furthermore, in 2020 the South African Broadcasting Corporation (SABC) also reported that almost 400 cases involving the SAPS members were being investigated and some of these cases include corruption, defeating the ends of justice, theft, fraud, and extortion [5].

All these reports indicate that an alternative mechanism is required that will ensure that criminal information and other digital evidence are well-secured. The Sunday Times reported that the ICDMS did not help to curb issues related to the loss or theft of case dockets of some of the high-profile criminal cases [1]. Note that this study views criminal information as any data that can be converted to digital data or evidence associated with a particular criminal case. Some of the issues associated with the availability of information lie in the mechanisms used by various parties to share criminal information because in most cases the applications used to share such information are designed in a centralised manner.

In some cases, the decentralised mechanism comes in using a gateway port that enables them to secretly share criminal information with interested parties outside their organisational boundaries. The use of such a gateway mechanism sometimes results in network traffic caused by broadcasting large volumes of data to a specific gateway port, which ends up straining the performance of the system (i.e., the system becomes offline).

Therefore, the primary problem of this study falls within the mechanisms used by various parties to either store or share criminal data with other parties that have an interest in it. For instance, the use of emails and other paper-based mechanisms to share criminal data might contribute towards illicit altering of criminal data for corruption purposes at any given stage. This might affect the fairness of the trial, including the execution of the court proceedings that determines the verdict. Hence, this study proposes a Blockchain-based model that might be used to securely share and preserve criminal information during its lifecycle by all the parties that are involved in the South African Criminal Justice System (SACJS). The novelty of this study lies in the securely and efficiently sharing of criminal data within the South African context, which includes the integration concept of Blockchain technology with the existing applications used by various parties within the SACJS.

The remainder of this study is structured as follows: Section II explores the research method adopted by this study. Section III details the background information associated with the criminal justice proceedings, including the SACJS and role players. Additionally, Section III also explores the background information related to the adopted technology, i.e. Blockchain. Section IV presents the proposed model that might be used to preserve criminal data as various parties securely share it. Section V presents the design of the proposed model. Thereafter, Section VI seeks to evaluate this study. Finally, the last section, which is Section VII, provides the concluding remarks of this study by summarising the study, as well as outlining some of the future work associated with this study.

II. Research methods

This study has adopted the following research methods to achieve the desired objectives, namely literature reviewing, modelling, and evaluation methodologies. A brief literature study is detailed to explore how criminal justice works, especially in the South African context. Additionally, a literature review was also used to explore how the adopted technology works, including some of the related work associated with this research. Note that this study makes use of a wide range of materials gathered from various sources such as documents or publications originating from the South African Government, newspaper articles, journal papers, conference papers, as well as content sourced from the internet. After exploring the literature review and identifying the gap, then this study proposes a model that can be used by various parties to share criminal information securely and efficiently within the SACJS. Hence, the modelling methodology is adopted to achieve this objective. The modelling methodology seeks to outline various components of the proposed model, which includes the proposed concepts that might

be used to integrate it with the existing systems used by various parties to interact with the criminal data. Thereafter, an evaluation is conducted to explore some of the benefits and shortcomings of the proposed solution.

III. Background

The details contained within this section are classified into two subsections namely: conceptualising the SACJS and adopted technology description. Conceptualising the SACJS subsection focuses on the background details of the South African justice processes and their role players, including the applications used by various role players. The adopted technology description subsection focuses on the background details of the technology solution adopted by this study, which is blockchain technology.

A. Conceptualising the South African criminal justice process

The SACJS consists of various processes and each of these processes is associated with a particular actor as shown in Figure 1. Note that the numbering from 1–8 in Figure 1 depicts various processes within the criminal justice system. As indicated in Figure 1, the criminal justice process starts when a crime is committed (represented by number 1) and reported to the SAPS (represented by number 2). Once the crime is reported, the SAPS will open a criminal case docket and assign it to an Investigating Officer for investigation. During the process of investigation, witnesses are identified, evidence is secured, and the accused person is also identified, as presented by number 3. Once the accused person is identified, an arrest is made by the SAPS which seeks to detain him or her, and, in some cases, the accused person might be released on bail, as depicted by number 4. However, note that the process of arresting or releasing the accused person also initiates other actors to initialise their mandate as required by the South African Constitution. Some of these actors that participate in such processes are the National Prosecuting Authority (NPA), Department of Justice & Constitutional Development (DJCD), Legal Aid South Africa (LASA), and the Department of Social Development (DSD). The NPA initialises the prosecution process. The DJCD initialises the court proceedings that seek to provide justice [6]. The LASA initialise the legal representation for individuals who cannot afford their legal representation in a court of law. Lastly, the DSD initialises the social support programmes for vulnerable people such as poor people, victims of crime, elderly people, and children.

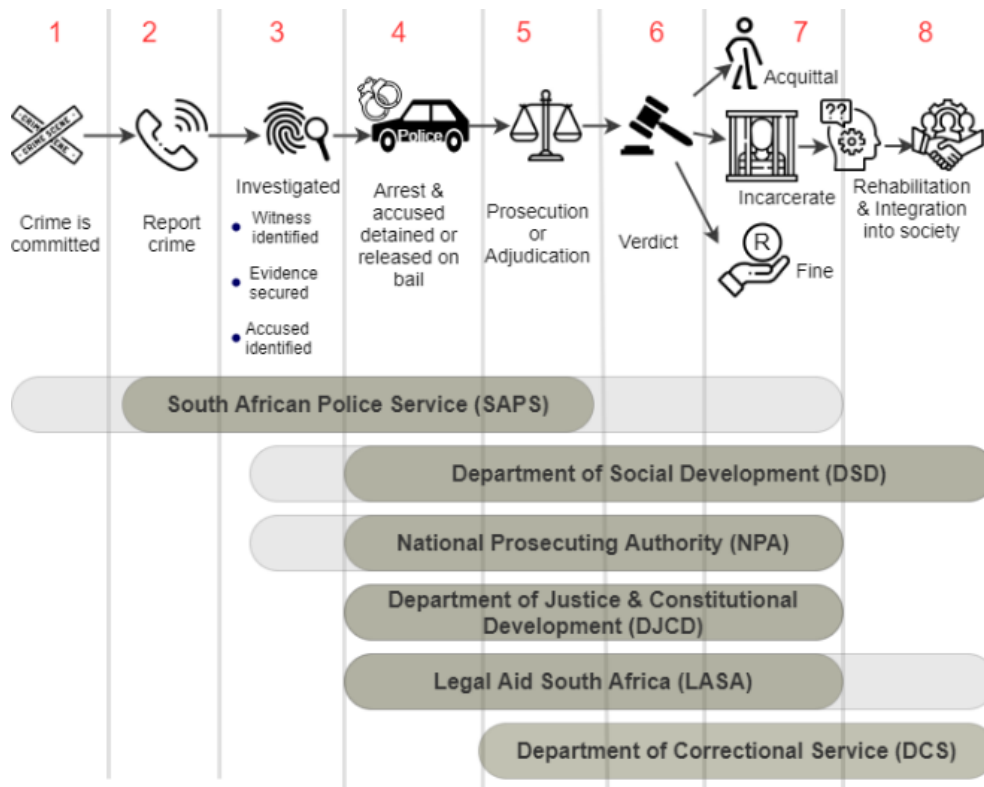


FIGURE 1: CRIMINAL JUSTICE PROCESS [7]

The process represented by number 5 details whether the accused person is prosecuted or in some cases, it might be an adjudication. Note that the NPA is responsible for taking the matter or case for prosecution if the evidence is strong enough to withstand the test of the court proceedings. This results in the handing over of the accused person to the Department of Correctional Service (DCS). The DCS is responsible for providing incarceration of inmates or detaining accused individuals as part of providing a rehabilitation facility and social reintegration of the offenders with their communities [8]. The process represented by number 6 depicts the verdict issued by the court of law, which might result in having three outcomes namely: *acquittal*, *incarceration*, or *fine*, as presented by number 7 of Figure 1. The *acquittal* process seeks to set free the accused person because the court has found him/her not guilty of the crime s/he has been charged with. The *incarceration* process seeks to detain the accused person after the verdict of the court found him/her guilty or when the accused person is regarded as a flight risk. The *fine* process seeks to release the accused with a penalty of paying a certain amount.

All the identified actors (i.e., SAPS, NPA, DJCD, LASA, and DCS) use various applications to accomplish some of their tasks. Therefore, the following items explore the various applications used by these actors to interact with criminal data:

- SAPS uses the *Integrated Case & Docket Management System* (ICDMS) to capture the details of the case docket and other forensic evidence.
- NPA uses the *Electronic Case Management System* (ECMS) to manage all the cases that are ready for prosecution.
- DJCD uses the *Integrated Case Management System* (ICMS) to provide accessible and quality justice for all by hearing the case evidence and deciding whether the accused is innocent or guilty.
- LASA uses the *Electronic Legal Aid Application* (eLAA) to assist those who cannot afford their legal representation.
- DSD uses the *Child Protection Register* (CPR) to ensure that the accused individuals that are younger than 18 years old obtain protection when it comes to detention.
- DCS uses the *Integrated Inmate Management System* (IIMS) to manage or record the details of the inmates or incarcerated individuals.
- Note that all these systems are integrated using the *Integrated Justice System* (IJS) as part of trying to share criminal information with all the parties that have an interest in the criminal data.

Figure 2 depicts the conceptualisation of how these various applications are integrated using IJS, to share criminal data among parties that have an interest in it. Note that the solid line depicted in Figure 2 represents the information flow used by the IJS, while the dotted lines represent the information flow of a respective application used by a specific actor or agent. Additionally, the IJS integration is based on the usage of a gateway portal to share criminal information.

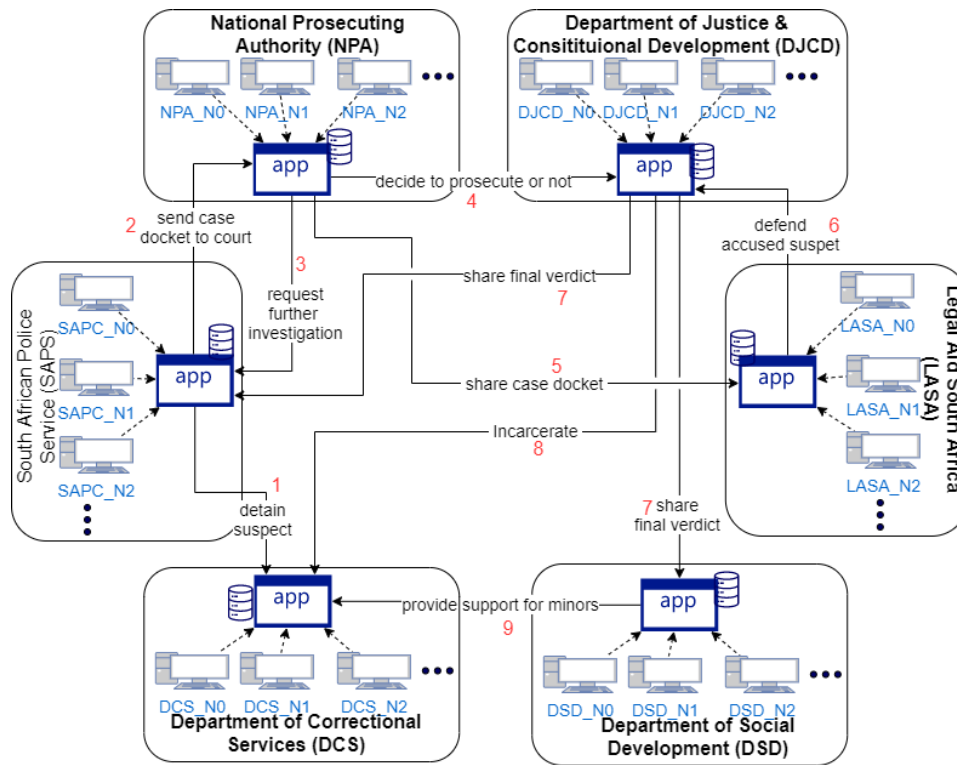


FIGURE 2: INTEGRATED JUSTICE SYSTEM CONCEPTS

B. Adopted technology description

Blockchain technology (BCT) is regarded as the technology used to implement a distributed and shared ledger system for storing a wide range of assets and transactions in a distributed manner. However, it is derived from a wider range of technologies that seek to use distributed ledger systems (DLSs) to store and share their information with various parties. Hence, the wider range of these technologies that use the DLS is referred to as distributed ledger technologies (DLTs), while BCTs are certain categories of DLTs that group their transactions into blocks, and these blocks of transactions are then linked together to form a chain-like data structure, hence, “Blockchain”. One of the benefits of using DLTs as a solution is that it automatically eliminates issues associated with a single point of failure experienced by centralised systems. Additionally, this also eliminates issues that emanate from having an actor that has central powers over criminal data, especially when that particular actor is reluctant to share it with others. For instance, when the prosecuting authority (from NPA) is reluctant to share information that implicates the accused individual with the legal representative (from LASA) of the accused person so that they can prepare for their own defense in the court of law. Another notable benefit of using the DLT solutions is that it enforces trust among the participants, not because they do not trust each other. However, they do not have to trust each other because all the nodes within the network have access to the same data since the identical copy of the ledger containing the

criminal data is replicated across multiple locations, regardless of their geographical location. Therefore, they will not have to worry about the illegal altering of criminal data that might be processed unnoticed. The use of DLS also promotes collaboration among various parties such as SAPS and NPA investigating officers because both of them will have access to the same data stored within the network.

The DLT solutions make use of cryptographic techniques to add or append new information to the network as part of trying to achieve the immutability of data across the network. However, the process of interacting with the information stored within the DLT solutions is governed by a smart-contract (SC) that is self-executing whenever the predefined conditions associated with particular transactions are met. All the participants (nodes) within the network make use of the consensus algorithms to either approve or synchronise the information stored with the network. The DLT solutions can be viewed as the solutions that seek to reduce risks, cost, and time when it comes to sharing criminal data among various parties. To achieve this objective, the proposed model makes use of the following mechanisms namely: cryptography, encryption, timestamping, DLS, and consensus algorithms, including ensuring that the shared data is immutable by default.

There are various blockchain frameworks available that can be adopted as a technology solution for this study, such as Bitcoin, Ethereum, Quorum, HydraChain, Hyperledger-Fabric (HLF), and MultiChain. However, some of these frameworks (i.e., Bitcoin, Ethereum, MultiChain, Quorum, and HydraChain) make use of cryptocurrency or mining algorithms to add new transactions to the network. The proposed solution seeks to store criminal data among various parties that are known to each other, therefore, a private blockchain that does not use any cryptocurrency or mining algorithms to add new transactions will be suitable for this study. Hence, HLF is a favorable framework adopted as a technology solution. HLF is a private blockchain framework that can be used to implement cross-industrial blockchain solutions [9].

All the participants of the HLF network are enrolled using a feature called Membership Service Provider (MSP). The DLS used by HLF is divided into two components namely: world-state (WS) and transaction log (TL). The WS stores the data that seek to describe the state of the DLS, while the TL seeks to record all the transactions that have resulted in the current state of the ledger. The Smart-contract (SC) is known as chaincode. HLF also makes use of a channel that can be used by various parties to secretly share information and each channel consist of peer nodes that contains the DLS used to share information within that channel. HLF uses the Raft

ordering service (which is also based on crash fault-tolerant) to collect all the transactions within the network and order them into blocks [10].

IV. A proposed Blockchain model for sharing criminal data

This section proposes a model that seeks to securely and efficiently share criminal information among various parties that have an interest in the criminal case. Figure 3 depicts an overview of the proposed model which is the share criminal evidence (ShareCrimE) model. The ShareCrimE model consists of four components:

- *Users/Agents* (represented by 1)– actors that play critical roles within the SACJS.
- *Application* (represented by 2)– mechanisms used by various actors to interact with the ShareCrimE model.
- *Services* (represented by 3)– mechanisms used by the applications to interact with the data stored within the blockchain network used by the ShareCrimE model.
- *Blockchain network* (represented by 4)– stores and distributes the criminal data among various nodes that form part of the ShareCrimE model.

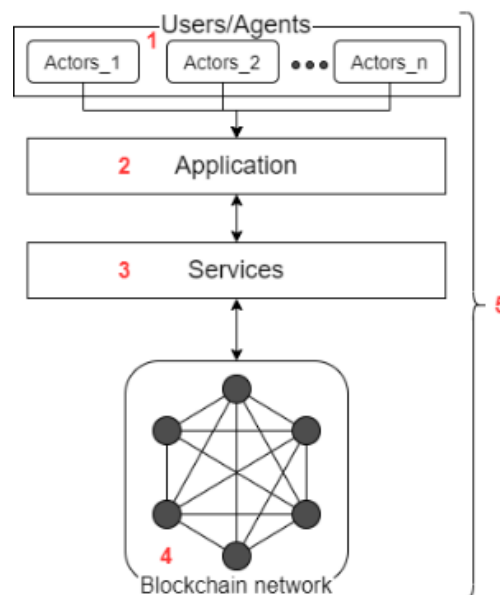


FIGURE 3: SHARECRIME MODEL OVERVIEW

This study adopts the following approach to explore the details of these components:

1. Identifying users/agents of the ShareCrimE model.
2. Establishing the application mechanisms that can be used by various users to interact with the criminal data.
3. Establishing the services offered by the ShareCrimE model.
4. Establishing the blockchain network used by the ShareCrimE model.

5. Defining the ShareCrimE model as an integration of the above steps, which are steps 1-4.

The following subsections explore the details of the adopted approach.

A. Identifying users/agents

The previous section has identified the SAPS, NPA, DJCD, LASA, DSD, and DCS as the agents of the SACJS. Note that this section only explores the roles of these actors that were not discussed in Section III to avoid repetition of some of the concepts that are already covered. Hence, the following items explore the roles of these actors within the ShareCrimE model:

1. SAPS– their role within the ShareCrimE model would be creating or scanning case dockets, appending digital evidence, sharing and accessing criminal data.
2. NPA– their role within the ShareCrimE model would be creating chargesheet, sharing, and accessing criminal data.
3. DJCD– their role within the ShareCrimE model would be creating verdicts, sharing, and accessing criminal data.
4. LASA– their role within the ShareCrimE model would be accessing criminal data and their verdicts, including sharing some of the data with NPA.
5. DSD– their role within the ShareCrimE model is to access the criminal data that can be used to identify the victims associated with a specific criminal case.
6. DCS– their role within the ShareCrimE model is to access criminal cases and their verdicts.

B. Establishing the application mechanisms

This component explores the mechanisms that might be used by actors to interact with criminal information. As indicated in Figure 4, all the applications used by various actors are integrated with the blockchain service Application Programming Interfaces (APIs) using features such as functions, logs, etcetera. One of the main reasons for integrating the proposed model with the existing systems is to avoid reinventing the wheel or developing a new system that will require more training on how it operates. The integration of the existing systems is aimed at securing the criminal information once it has been captured by the current systems used by these various actors. Note that, the criminal information stored within the proposed model can be trusted because it stores immutable data by default, which implies that this information cannot be changed or altered for corruption purposes. Hence, the application layer depicts the use of APIs to integrate the existing systems with the blockchain as the underlying technology solution.

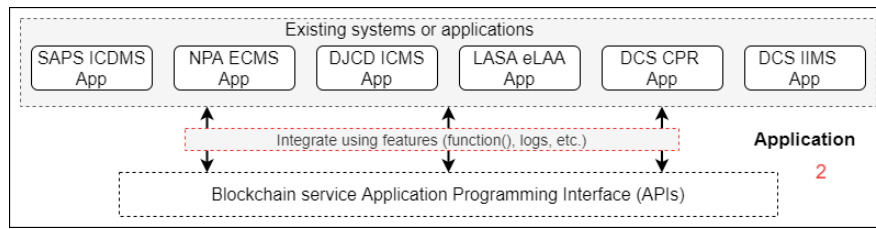


Figure 4: ShareCrimE model: Application layer

C. Establish the services mechanisms

This section explores the details that seek to manage how the application layer (represented by number 2 in Figure 3) interacts with the blockchain network layer (represented by number 4 in Figure 3). This study classifies the services used by HLF to manage the interaction between the application layer and the blockchain network layer into three categories namely: identity management, wallet management, and network gateway. The following items explore the details of these three categories:

- *Identity management*– is used for managing the identities of various resources (such as nodes, applications, and administrators) within the blockchain network [11]. Hence, each of these resources must be associated with a specific digital identity or certificate that can be used by the blockchain network to determine how they access certain resources or information within the network. Note that all these identities are issued by a trusted authority and HLF achieves this by using an MSP. Additionally, the MSP uses the X.509 certificates as identities that rely on the Public Key Infrastructure (PKI) hierarchical model [11].
- *Wallet management*– is used for managing the identities of various actors (which is the organisation and its members) that seek to participate within the blockchain network [12]. Note that this mechanism is embedded within an application used by these actors as they interact with the blockchain network. However, the process of it starts when a user logs in to an application and uses their credentials to connect with the blockchain network using a specific channel associated with relevant identities. A channel can be viewed as a private blockchain overlay that enables various participants to secretly share information.
- A *network gateway*– is used for managing the interactions between the blockchain network and applications used by various actors [13]. Note that applications use a connection profile to configure a gateway that seeks to handle its interactions because it describes a set of components associated with various nodes (i.e., peer nodes and ordering nodes), and certificate authorities (CAs) [14]. Additionally, the connection profile contains the channel

and participants' (i.e., organisations') information that uses these components [14]. The role of CA within the network is to issue the PKI-based certificates to the participants (i.e., organization) and their members.

D. Blockchain network

Figure 5 depicts the blockchain network used by the ShareCrimE model, which consists of the following components: *CAs*, *Raft ordering service*, *channels*, and *peer nodes*. The following items explore the details of each of these components as identified in Figure 5:

- *A CA*– the ShareCrimE model consists of 6 CAs namely: SAPS_CA, NPA_CA, DJCD_CA, LASA_CA, DCS_CA, and DSD_CA, as shown in Figure 5.
- *Raft ordering service*– the ShareCrimE model consists of 6 ordering nodes represented by node_1 to node_6 in Figure 5.
- *Channels*– the ShareCrimE model consists of 7 channels, 6 of these channels are associated with a respective organisation (i.e., SAPS, NPA, DJCD, LASA, DCS, and DSD), while the 7th channel is used as the main channel that enables these organisations to share criminal information outside their boundaries (represented by ShareCrimE main channel), as shown in Figure 5. Note that all these channels make use of the same Raft ordering service to group their transactions into blocks and distribute the blocks to relevant peer nodes that are associated with a particular channel.
- *Peer nodes*– each organisation consists of 6 peer nodes, whereby all of them are connected to their respective channels, and 3 of which are connected to the main channel. This results in having 18 peer nodes connected to the main channel. Note that each of these peer nodes contains the SC and the DLS of their respective channels. For instance, node SAPS_P1 consists of two SCs (one for the SAPS channel and the other one for the main channel) and two DLSs (one for the SAPS channel and the other one for the main channel). Nodes SAPS_P4 to SAPS_P6 each contains only one SC and DLS. Similarly, this setup applies to all other organisations.

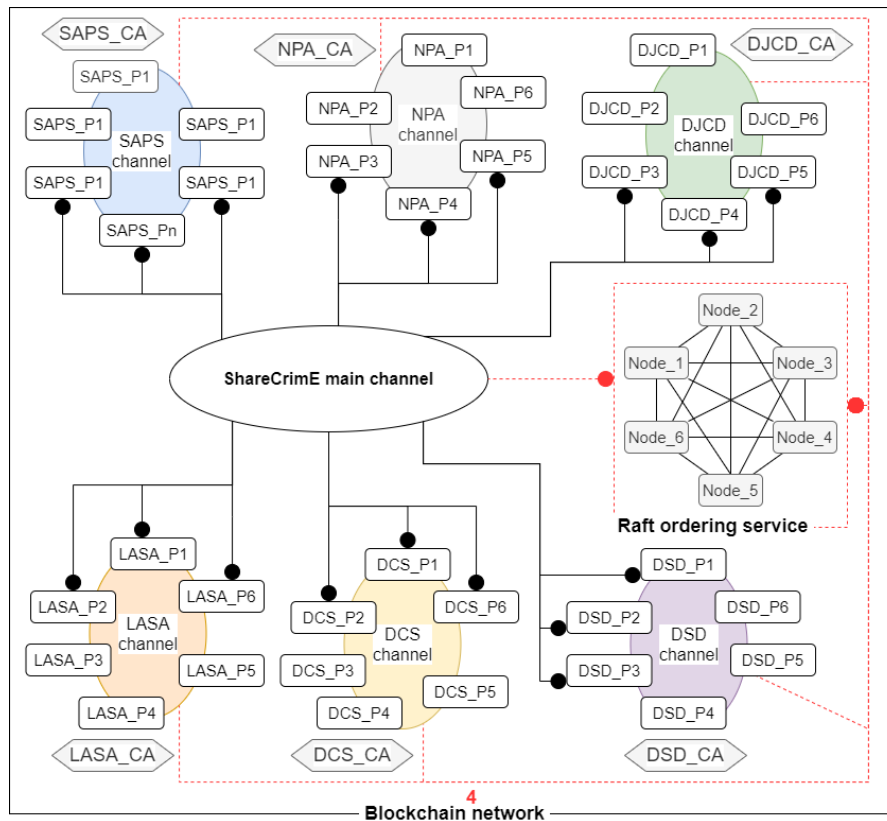


FIGURE 5: SHARECRIME BLOCKCHAIN NETWORK

The following section seeks to integrate steps 1-4 (as shown in Figure 3) to generate the final step which is defining the ShareCrime model (represented by step 5 in Figure 3).

E. ShareCrime high-level model

Figure 6 depicts a high-level of the ShareCrime model as an integration of all the other steps, which are steps 1-4 in Figures 3 and 6. The information flow starts when various actors submitting criminal data using various applications. Thereafter, the ShareCrime model is embedded within these applications using various mechanisms such as features or functions. All the identities of the actors that seek to access data stored within the ShareCrime model are assessed by various services to preserve confidentiality, data integrity, and availability. The blockchain network used by the ShareCrime model comprises of the following components namely: certificate authority, raft ordering services, and peer nodes, as shown in Figure 6.

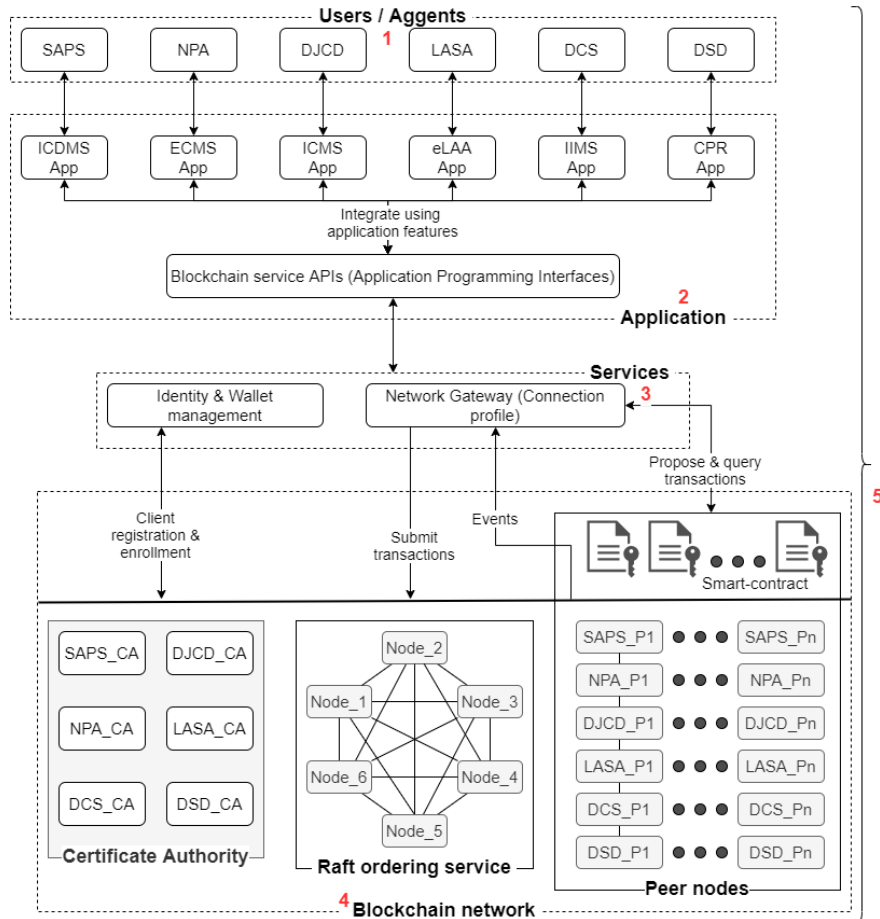


FIGURE 6: SHARECRIME HIGH-LEVEL MODEL

This section has presented the proposed high-level model that can be used to share criminal data securely and efficiently. Therefore, the following section explores the design of the ShareCrimE model.

V. ShareTendPro model design

The ShareCrimE model design represents a detailed view of the model and is classified into three categories, namely: overview of the ShareCrimE model information flow, the sequence diagram of the ShareCrimE Application channel, and the sequence diagram of the ShareCrimE main channel.

A. ShareCrimE model information flow

Figure 7 depicts an overview information flow of the ShareCrimE model as part of trying to visualise how the proposed model process or stores criminal data. Additionally, the information flow also depicts how various components interact with the criminal data, which includes how digital forensic evidence can be accessed within the proposed model. The information flow starts when the participants submit their transactions to the blockchain network used by the

ShareCrimE model. Once the submitted transactions meet all the predefined conditions stipulated on the SC associated with a specific communication channel, then such transactions will be accepted by the ShareCrimE network. Thereafter, the ordering service will collect all the accepted transactions and group them into a block and distribute these blocks to all the nodes associated with that channel. As indicated earlier on, the DLS consists of two components namely: TL and WS. The TL in this case, stores the digital forensic data that can be used by forensic investigators to analyse what has transpired within the criminal case, while the WS stores the actual data that has resulted in the current state of the blockchain network. Note that the TL can also be used by other participants to verify or check what has transpired within a particular crime case of their interest. Again, note that this functionality is only available to all the authorized parties such as lawyers or judges.

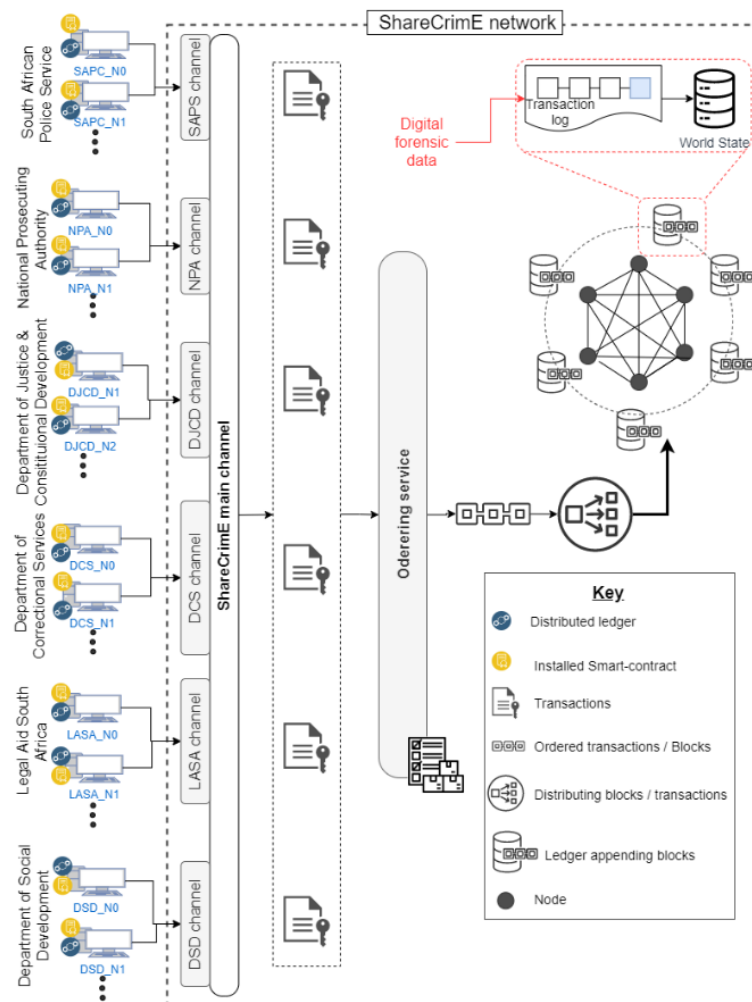


FIGURE 7: SHARECRIME MODEL INFORMATION FLOW OVERVIEW

B. Sequence diagram of the ShareCrimE Application channel

Figure 8 depicts a sequence diagram associated with an application channel (App channel). The App channel in this case refers to the channels used by various participants to share criminal

data within their organisational boundaries. For instance, the organisational boundaries of the SAPS channel lie in sharing of criminal information among all the members of the SAPS regardless of their geographical location. Therefore, when SAPS members use the ICDMS application to *add*, *update* or *delete* criminal data then such information will be sent to the SC and their transactions will then be grouped into blocks using the Raft ordering service. Thereafter, the blockchain network will then record all the activities associated with the operations that seek to create, modify, or delete criminal data within the network. Once the block of transactions is added or appended to the blockchain network then such information is distributed among all the members of the SAPS to append their ledger. Note that this notion can also be applied to other App channels of the following participants (NPA, DJCD, LASA, DCS, and DSD). Hence, this study will not explore them to avoid repetition of similar concepts.

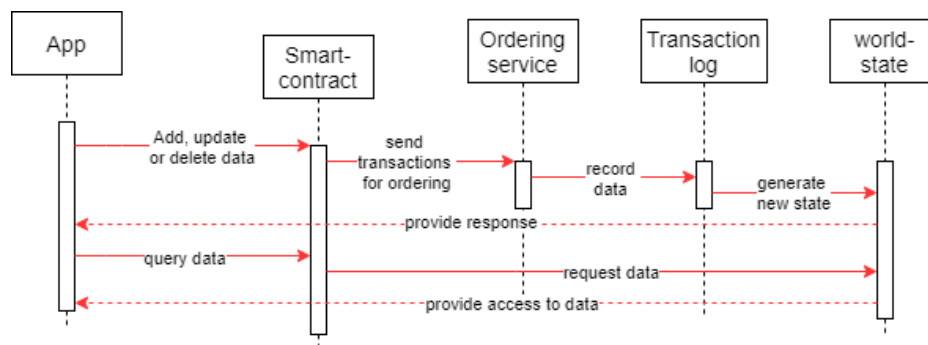


FIGURE 8: SEQUENCE DIAGRAM OF APP CHANNEL

C. Sequence diagram of the ShareCrimE main channel

Figure 9 depicts a sequence diagram associated with the main channel used by the ShareCrimE model. This channel seeks to share criminal data among various applications used by different participants (i.e., organisations). The SAPS App in this case will share the criminal data associated with the arrests with all the participants. The process of sharing such criminal information start when the SAPS App seeks to create or share criminal evidence of the accused person. This information will then be assessed to whether it met all the predefined conditions stipulated on the SC, and once the assessment is complete all the accepted transactions associated with the arrest are grouped into blocks (represented by A1) and distributed to all the nodes within the network. The records of these blocks of transactions are stored in the TL (represented by A2), while the actual data is stored in the WS (represented by A3). Thereafter, the NPA will then share the prosecution details associated with that criminal case, whereby the recording or storing of evidence is handled the same way as depicted within the SAPS App (represented by A) by the blockchain network. Once the prosecution details are successfully

captured within the blockchain network, then such information is distributed to all the parties that form part of the blockchain network. After sharing the prosecution details with relevant parties, the DJCD shares their verdict by submitting it to the SC and then the blockchain network handles another process of storing data or digital evidence including distributing these details to other parties. Once the verdict of the DJCD is associated with incarceration, then the details of the arrested person will be shared with the DCS detailing the sentence of that person. The DCS will then share their records with all their members and the SAPS as part of keeping the records of their inmates.

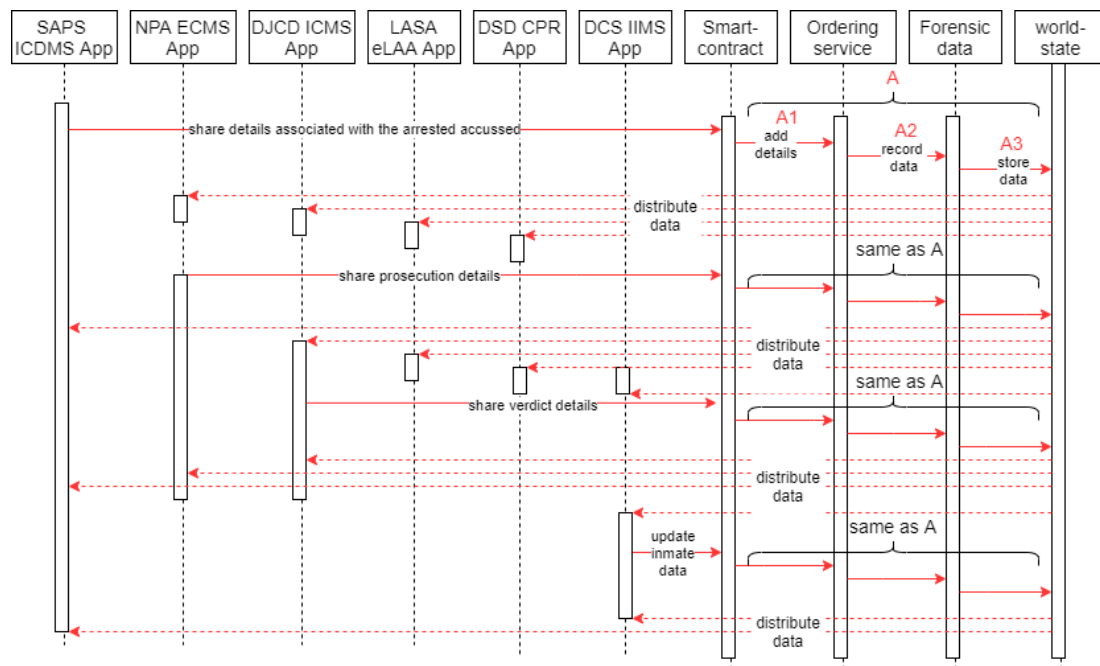


Figure 9: sequence diagram of the main channel

VI. Evaluation

The details contained within this section seek to explore the benefits and shortcomings of this research study.

A. Benefits of the research

Various benefits can be drawn from this study. The following items explore the detail of these benefits:

1. Enhanced information security– refers to the mechanisms used to secure criminal data from unauthorised parties to either access or tamper with it. Some of the mechanisms used to secure this data are storing it in multiple locations, using cryptography to encrypt data, the use of secure communication channels, the use of timestamps, and having immutable transactions for data integrity.

2. Greater transparency– is derived from the distributed nature of the proposed model because the business process used to either create, modify, or delete criminal data is transparent. Hence, the authorised parties will not have to worry about having criminal data that will be processed unnoticed, which disables the rogue parties to collude over criminal data since all parties will have access to the entire history of the criminal data.
3. Time efficiency– ensures that all the parties within the proposed model will have access to the entire history of the criminal data of their interest. Note that this data is stored within the TL since it seeks to record all the activities associated with the following operations: create, update, or delete criminal data within the network. Therefore, this data is readily available to all the authorised parties that form part of the network that seeks to determine what has transpired within that data. For instance, having access to such information by the NPA will enable them to access the criminal case and draught the chargesheet in real-time which reduces the time taken to wait for the case docket to be completed.
4. Credible evidence– refers to the digital data that can be presented or accepted by the court of law as evidence of what has transpired within that criminal case. Hence, having access to the entire history of the criminal data by either SAPS, NPA, DJCD, or LASA presents them with a chronological sequence used to either handle or share the criminal data. This ensures that the integrity of the data stored within the network has not been tampered with. Some of the mechanisms used by the blockchain network to secure such critical evidence are timestamp, cryptographical encryption, and immutable data that cannot be altered for corruption purposes.
5. Promotes real-time investigations– the distributed nature of the proposed model will enable parallel investigations by both the SAPS and NPA as part of gathering more evidence that can be corroborated by the criminal case. This benefit will also assist when it comes to delays in case dockets since the NPA will be able to determine whether a particular case docket is ready for a court trial or not. Additionally, having access to criminal data allows all the parties to process their reports quickly, which can also promote having a quicker delivery of justice or judgment associated with a specific criminal case.
6. Enforces trust– the proposed model inherently has trust built into the blockchain network since it consists of the mechanisms that seek to promote transparency, accountability, data integrity, data confidentiality, and secure criminal data by having immutable data that depicts what has transpired within the criminal case. Hence, all the parties will have more trust in the criminal data without having to worry about the illegal altering of data that might be processed unnoticed.

7. Improve collaboration– the proposed model will benefit various parties that seek to achieve the same objectives, especially when it comes to investigation processes whereby more investigators from different locations are investigating the same suspect. Additionally, it also promotes collaboration among the SAPS and NPA when it comes to criminal cases that require further investigations.

B. Shortcomings of the research

1. Integration issues– this challenge is anticipated because some of these applications used by various parties to share criminal information are rendered as a service by third parties to relevant participants that form part of the network. Hence, it might be difficult to integrate the proposed model with some of the existing applications.
2. Lack of political will– this can only be regarded as a shortcoming because most of the high-ranking positions within these organisations can be influenced by political will. Hence, some of the people holding these positions might be reluctant towards the adoption of the proposed solution because they are involved in some of the corrupt activities, or they are associated with some of the corrupt individuals.

VII. Related work

The study performed by Lone and Mir [15] proposes a blockchain-based forensic chain of custody that can be used to address issues related to data integrity within the forensic space. However, their proposed solution is based on Hyperledger-composer (HLC) which is a deprecated tool. The following study [16] also uses HLC to either propose or implement concepts that can be associated with digital forensics, a chain of custody, and information sharing. The studies of [15, 17, 18, 19, 20] adopt a different approach and use the Ethereum framework to either propose or implement their concepts or prototypes. The Ethereum framework makes use of native cryptocurrency and mining algorithms to add new transactions to the network. The proposed model presented by this study is similar to the studies by [15, 18, 21, 22] because they also seek to propose a blockchain model that can be used to address issues associated with the criminal justice system. The model presented by [21] is implemented using Hyperledger-sawtooth (HLS) which is based on the following consensus algorithms: Practical Byzantine Fault Tolerance and Proof of Elapsed Time [23]. The HLS supports both private and public Blockchain solutions, however, it relies on a third-party organisation, Intel, since it uses Software Guard Extensions [24]. Therefore, Table 1 seeks to compare related work based on whether the identified feature is favourable or not.

Table 1: comparing the related work with the ShareCrimE model

Features of the ShareCrimE model	[15]	[18]	[21]	[22]	ShareCrimE model
Support private Blockchain	√	√	√	√	√
Does not rely on cryptocurrency or mining algorithm to add new transactions to the network			√		√
Support integration of the existing systems					√
Support more actors in the criminal justice process	√	√	√	√	√
Based on the SACJS					√
Share criminal information	√	√	√	√	√
Does not rely on a third-party organization	√	√		√	√
Adopts the use of HLF					√

The ShareCrimE model accommodate all the identified features highlighted in Table 1. However, note that the following features, i.e., support integration of the existing systems, based on SACJS, and adopts the use of HLF were not explored by any of the related work. Additionally, only one of the related works explored the feature: “Does not rely on either cryptocurrency or mining algorithms to add new transactions to the network”. The other features are accommodated by most of the related work.

VIII. Conclusion

The ShareCrimE model has demonstrated how blockchain technology might be adapted to securely share criminal data among various parties that have an interest in it. The adoption of the ShareCrimE model will promote greater transparency and accountability since various parties will not be able to collude over criminal data. Additionally, it will also promote justice to all the affected parties because the data stored within the proposed mode is immutable by default, which means it cannot be deleted for corruption purposes. The proposed model also enforces trust over criminal data since parties will not have to worry about having illegal altering of criminal data processing going unnoticed. Furthermore, the ShareCrimE model ensures that all the parties that have an interest in the criminal data have instant access to real-time data stored within the blockchain network, which might help them to process their reports faster.

The ShareCrimE mode will provide a revolutionary step towards curbing corruption taking place within the criminal justice system that is largely influenced by corruption activities. However, note that the notion applied within this study might also be adapted to other countries that face similar issues or that seek to share criminal data securely and efficiently.

In the future, this study will focus on implementing the proposed model to come up with a proof of concept.

Acknowledgment

This research study was funded and supported by the Council for Scientific and Industrial Research (CSIR) and the University of Pretoria (UP), South Africa. Special thanks go to Prof H. Venter (UP) and Mr. W.H. le Roux (CSIR) for their continuous support and contribution to the success of this research.

References

- [1] SundayTime, "A hole in Pistorius conman case docket," TimesLive News , 23 March 2016. [Online]. Available: <https://www.timeslive.co.za/news/south-africa/2016-03-23-a-hole-in-pistorius-conman-case-docket/>. [Accessed 28 March 2022].
- [2] EyeWitness, "DA demands in-field training for SAPS officers in the Western Cape," EyeWitness, March 2022. [Online]. Available: <https://ewn.co.za/2022/03/04/da-demands-in-field-training-for-saps-officers-in-the-western-cape>. [Accessed 28 March 2022].
- [3] South African Government, "MEC Albert Fritz on e-docket software not being effectively used by SAPS and Courts," Government speeches, 5 March 2020. [Online]. Available: <https://www.gov.za/speeches/r6135-million-e-docket-software-not-being-effectively-used-saps-courts-%C2%A0-5-mar-2020-0000>. [Accessed 28 March 2022].
- [4] Carte Blanche, "Dockets for sale | Carte Blanche | M-Net," Carte Blanche youtube channel, 11 April 2022. [Online]. Available: <https://www.youtube.com/watch?v=s1L8j50Sgz0>. [Accessed 19 April 2022].
- [5] SABC news, "Almost 400 corruption cases involving SAPS members being investigated," SABC news, 31 November 2020. [Online]. Available: <https://www.sabcnews.com/almost-400-corruption-cases-involving-saps-members-being-investigated/>. [Accessed 28 March 2022].
- [6] South African Government, "Strategic Plan for the period 2011-2016: Annual review 2011/12," South African Government, March 2011. [Online]. Available: https://www.gov.za/sites/default/files/gcis_document/201409/mtsf0.pdf. [Accessed 25 April 2022].
- [7] South African Parliamentary Monitoring Group , "Prgress Report: Integrated Justice System (IJS) Programme," Parliamentary Reports, 31 May 2017. [Online]. Available: <https://static.pmg.org.za/170531IJSReport.pdf>. [Accessed 24 March 2022].
- [8] Department of Correctional Services, "Mission/ Vision/Values," www.dcs.gov.za, 2022. [Online]. Available: http://www.dcs.gov.za/?page_id=174. [Accessed 20 April 2022].
- [9] Hyperledger Fundation, "About Hyperledger Foundation," The Linux Foundation Projects: Hyperledger, 2021. [Online]. Available: <https://www.hyperledger.org/about>. [Accessed 28 03 2022].

- [10] Hyperledger-fabric 2.2, "Pluggable consensus," Hyperledger-fabric document, 2021. [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/whatis.html?highlight=BFT#pluggable-consensus>. [Accessed 23 May 2022].
- [11] Hyperledger-fabric, "Identity," Release-2.2, 2021. [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/identity/identity.html?highlight=certificate%20authority#what-is-an-identity>. [Accessed 25 05 2022].
- [12] Hyperledger-fabric, "Wallet," Release-2.2, 2022. [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/developapps/wallet.html>. [Accessed 25 05 2022].
- [13] Hyperledger-fabric, "Gateway," Release-2.2, 2021. [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/developapps/gateway.html>. [Accessed 25 05 2022].
- [14] Hyperledger-fabric, "Connection Profile," Release 2.2 document, 2021. [Online]. Available: <https://hyperledger-fabric.readthedocs.io/en/release-2.2/developapps/connectionprofile.html>. [Accessed 25 05 2022].
- [15] A. Lone and R. Mir, "Forensic-chain: Blockchain based digital forensics chain of custody with PoC in Hyperledger Composer," *Digital investigation* 28, pp. 44-55, 2019.
- [16] H. Elgohary, S. Darwish and S. Elkaffas, "Improving Uncertainty in Chain of Custody for Image Forensics Investigation Applications," *IEEE Access* 10, pp. 14669-14679, 2022.
- [17] L. Ahmad, S. Khanji, F. Iqbal and F. Kamoun, "Blockchain-based chain of custody: towards real-time tamper-proof evidence management," *In Proceedings of the 15th International Conference on Availability, Reliability and Security*, pp. 1-8, 2020.
- [18] M. Li, C. Lal, M. Conti and D. Hu, "LEChain: A blockchain-based lawful evidence management scheme for digital forensics," *Future Generation Computer Systems* 115, 2021.
- [19] E. Yuniato, Y. Prayudi and B. Sugiantoro, "B-DEC: digital evidence cabinet based on blockchain for evidence managemen," *International Journal of Computer Applications*, vol. 975, 2019.
- [20] S. Bonomi, M. Casini and C. Ciccotelli, "B-coc: A blockchain-based chain of custody for evidences management in digital forensics," in *arXiv preprint arXiv:1807.10359*, 2018.
- [21] A. Khan, M. Uddin, A. Shaikh, A. Laghari and A. Rajput, "MF-ledger: blockchain hyperledger sawtooth-enabled novel and secure multimedia chain of custody forensic investigation architecture," *IEEE Access*, vol. 9, pp. 103637-103650, 2021.
- [22] F. F. Alruwaili, "CustodyBlock: A Distributed Chain of Custody Evidence Framework," *Information* 2021, vol. 12, no. 2, p. 88, 2021.
- [23] Hyperledger, "Type: Distributed ledger software," Hyperledger-Sawtooth, 2022. [Online]. Available: <https://www.hyperledger.org/use/sawtooth>. [Accessed 25 05 2022].
- [24] S. Kaur, S. Chaturvedi, A. Sharma and J. Kar, "A research survey on applications of consensus protocols in blockchain," *A research survey on applications of consensus protocols in blockchain*, 2021.