**CROSS-ENTERPRISE DOCUMENT SHARING (XDS) IMPLEMENTATION BASED ON BLOCKCHAIN TECHNOLOGY**

**PETNATHEAN JULLED**

**A THESIS SUBMITTED IN PARTIAL FULFILLMENT**

**OF THE REQUIREMENTS FOR**

**THE DEGREE OF MASTER OF SCIENCE**

**(CYBER SECURITY AND INFORMATION ASSURANCE)**

**FACULTY OF GRADUATE STUDIES**

**MAHIDOL UNIVERSITY**

**2019**

**COPYRIGHT OF MAHIDOL UNIVERSITY**

Thesis

entitled

**CROSS-ENTERPRISE DOCUMENT SHARING (XDS) IMPLEMENTATION BASED ON BLOCKCHAIN TECHNOLOGY**

|  |  |
| --- | --- |
|  | …………………………………….......  Mr. Petnathean Julled,  Candidate |
|  | …………………………………….......  Assadarat Khurat,  Dr. –Ing. (Computer Security)  Major advisor  …………………………………….......  Pattanasak Mongkolwat,  Co-advisor  …………………………………….......  Asst. Prof. Thitinan Tantidham,  Dr.rer.nat. (Computer Science)  Co-advisor |
| …………………………………….......  Pattanasak Mongkolwat,  M.D., Ph.D. (Biochemistry)  Dean  Faculty of Graduate Studies  Mahidol University | …………………………………….......  Assoc. Prof. Vasaka Visoottiviseth,  Ph.D. (Computer Engineering)  Program Director  Master of Science Program in Cyber Security and Information Assurance  Faculty of Information and Communication Technology  Mahidol University |

Thesis

entitled

**CROSS-ENTERPRISE DOCUMENT SHARING (XDS) IMPLEMENTATION BASED ON BLOCKCHAIN TECHNOLOGY**

was submitted to the Faculty of Graduate Studies, Mahidol University

for the degree of Master of Science (Cyber Security and Information Assurance)

on

May 31, 2021

|  |  |
| --- | --- |
|  | …………………………………….......  Mr. Petnathean Julled,  Candidate |
|  | …………………………………….......  Assadarat Khurat,  Dr. –Ing. (Computer Security)  Major advisor  …………………………………….......  Pattanasak Mongkolwat,  Co-advisor  …………………………………….......  Asst. Prof. Thitinan Tantidham,  Dr.rer.nat. (Computer Science)  Co-advisor |
| …………………………………….......  Pattanasak Mongkolwat,  M.D., Ph.D. (Biochemistry)  Dean  Faculty of Graduate Studies  Mahidol University | …………………………………….......  Assoc. Prof. Vasaka Visoottiviseth,  Ph.D. (Computer Engineering)  Program Director  Master of Science Program in Cyber Security and Information Assurance  Faculty of Information and Communication Technology  Mahidol University |

**ACKNOWLEDGEMENTS**

The success of this thesis can be succeeded by the attentive support from …………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………….

I would like to thank the entire respondent who was the sampling in this study…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

Petnathean Julled

|  |
| --- |
| CROSS-ENTERPRISE DOCUMENT SHARING (XDS) IMPLEMENTATION BASED ON BLOCKCHAIN TECHNOLOGY  PETNATHEAN JULLED 5936474  M.Sc. (CYBER SECURITY AND INFORMATION ASSURANCE)  THESIS ADVISORY COMMITTEE: ASSADARAT KHURAT, Ph.D.,  PATTANASAK MONGKOLWAT, Ph.D., THITINAN TANTIDHAM, Ph.D.  ABSTRACT  On the increasing demand for better quality of healthcare service, there is the topic that involve healthcare information technology in term of operation efficiency. Healthcare information sharing and interoperability between healthcare organizations is one of major solution to improve healthcare service quality. But, there still many challenge inhibit the solution to become reality. There found initiatives to standardize healthcare information sharing method. To address issue about health document sharing between different enterprises, Integrating Healthcare Enterprise (IHE) initiative have proposed Cross-Enterprise Document Sharing (XDS.b) Profile. The profile allow the adopted organizations to share health document between each other simultaneously.  As well as other industry, there also emerging cyber-security threats threatening healthcare information domain. These threats increase difficulty to development of health information sharing network and causing damage to healthcare enterprises. These cyber-threats can cause damage to the industry in many aspect, especially those cyber-attack that targeting integrity and availability of data. These kind of cyber-attack can severe the continuity of medical operation which potentially can result as the cost of patient’s life. There are many solutions technology proposed to deal with these kind of cyber-attacks. One of the technology that on the trend to deal with cyber-threats threatening integrity and availability of data is Blockchain technology.  There are several researches and concepts that proposed about using Blockchain technology to solve health information sharing issue. But there still many limits prevent Blockchain technology to effectively integrated with data like health information. In this work, we propose another approach for integrate Blockchain technology with health information. We see that standard like IHE XDS.b profile could be use with Blockchain technology to allow health document sharing through decentralized network while address cyber-security issue through unique characteristics of Blockchain technology.  KEY WORDS: HEALTH INFORMATION / INTEROPERABILITY / INFORMATION SHARING / INFORMATION SECURITY / BLOCKCHAIN / SMART CONTRACT / IHE / XDS  40 pages |

**CONTENTS**

|  |  |
| --- | --- |
|  | Page |
| **ACKNOWLEDGEMENT** | iii |
| **ABSTRACT** | iv |
| **LIST OF TABLES** | v |
| **LIST OF FIGURES** | vii |
| **LIST OF ABBREVIATIONS** | ix |
| **CHAPTER I INTRODUCTION** | **1** |
| 1.1 Motivation……………………………………………………………… | **Error! Bookmark not defined.** |
| 1.2 Problem Statement…………………………………………………....... | 4 |
| 1.3 Objective……………………………………………………………….. | 4 |
| 1.4 Scope of Project……………………………………………………....... | 5 |
| **CHAPTER II LITERATURES REVIEW** |  |
| 2.1 State of Cyber Security and Cyber Threats in Healthcare Domain….… |  |
| 2.2 Integrate the Healthcare Enterprise (IHE) ………… |  |
| 2.3 Blockchain Technology………………………………………………... |  |
| 2.4 Related Work…………………………… |  |
| **CHAPTER III METHOD** |  |
| 3.1 A Use Case Scenario ……………………………………. |  |
| 3.2 Network Design …………………………………... |  |
| 3.3 Blockchain Components and Consensus ………….. |  |
| 3.4 Establish foundation of trust amongst the network |  |
| 3.5 Integrating Blockchain with XDS.b |  |
| 3.6 XDS Blockchain and Smart-contract Design |  |
| 3.7 An augmentation to mitigate data-corruption incident |  |
|  |  |
| **CHAPTER IV IMPLEMENTATION** |  |
|  |  |
| Acquiring IHE ITI transaction samples and analysis |  |
| XDS Document Metadata attributes analysis |  |
|  |  |
|  |  |
| 4.1 Blockchain setup  -> subsection: Geth + Client  -> subsection: Utilize Quorum |  |
|  |  |
|  |  |
| 4.2 XDS Actors  Intro. paragraph 1: utilize Javascript with NodeJS, paragraph 2: utilize Web3JS (references attached) 4.2.1 Registry Actor   - Interpreter  - Implementing Document Registry Smart Contract 4.2.2 Repository Actor 4.2.3 Consumer Actor  Implementing Document Search function over smart-contract  Interprete XML messages with xml2js module |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| **CHAPTER V RESULT** |  |
| 5.1 Implementation Summary |  |
| 5.2 Performance Evaluation Result |  |
| **CHAPTER VI CONCLUSION** |  |
| 6.1 Answering Problem Statement |  |
| 6.2 Implementation Source Code |  |
| **CHAPTER VII DISCUSSION** |  |
| 7.1 Limit of implementation based on current technology |  |
| 7.2 Suggestion for concept adoption |  |
| 7.3 Suggestion for future work |  |
| **REFERENCE** |  |

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **Table** |  | **Page** |
| 1 | SubmissionSet Metadata Attributes | 13 |
| 2 | Folder Metadata Attributes | 14 |
| 3 | DocumentEntry Metadata Attributes | 14 |

**LIST OF FIGURES**

**ตัวอย่างหน้าบทคัดย่อภาษาอังกฤษ**

|  |  |  |
| --- | --- | --- |
| **Figures** |  | **Page** |
| 5.17 | Cytotoxicity screening of mangosteen extract at various concentrations on SK-N-SH cells at 24 hours incubation by MTT assay | 72 |

**LIST OF ABBREVIATIONS**

**ตัวอย่างหน้าบทคัดย่อภาษาอังกฤษ**

|  |  |  |
| --- | --- | --- |
| **Abbreviation** |  | **Page** |
| IHE | Integrating Healthcare Enterprise | 1 |
| XDS | Cross – Enterprise Document Sharing | 1 |
| XDS.b | Cross – Enterprise Document Sharing Set-b | 1 |

# 

**CHAPTER I**

**INTRODUCTION**

With the transition from the age of paperwork, the healthcare industry is now undergoing digital transformation. Efficiency and continuity are the main factors that driven the healthcare industry to change. Paperwork starts falling behind when a huge amount of data is produced by healthcare service operation from day to day. Health information undeniably becoming an important component in developing efficient healthcare services [1–6]. On the increasing demand for the better quality of healthcare service, there is the topic that involves healthcare information technology in terms of operational efficiency. Healthcare information sharing and interoperability between healthcare organizations are one of the major solutions to improve healthcare service quality. Patient’s health document data are scattered across different healthcare organizations, due to the foundation of healthcare informatics are separately developed by different organizations. Each healthcare organizations have their own method to process and handle healthcare information. This makes it hard for one healthcare piece of information to interoperate with other. To enable health information sharing from just one organization with one another can cost much more than the benefit they can gain. This even did not regard concern about business value. Sharing health information with not fully-trusted party exposing vulnerabilities to the business model. The risk that benefits the organization gain from sharing their patient information with others may not sustain the risk and cost they need to take. This creates high friction for one organization to share their information with others. It was even more difficult for an individual patient to integrate their healthcare with different providers. It revealed that these interoperation problems cause a huge decrease inefficiency in healthcare operation and result in lower quality of healthcare service [7–14].

That way many initiatives start to standardize healthcare information technology to allow healthcare organizations to be able to interoperate with each other. Integrating Healthcare Enterprise (IHE) is one of the well-known initiatives that provide materials for healthcare informatics standardization. IHE provides an implementation framework and guideline for developing a health informatics system. For health document sharing between different organizations, they provide a Cross-Enterprise Document Sharing (XDS.b) profile. The profile act as a guideline for the system developer to implement their system to meet the requirement where the system can share health document with other organizations. This profile will be the main tool for this work, to deal with the health information sharing problem.

Motivation

In the current age of information digitalization, cybersecurity has become an issue for many organizations and individuals. Anyone can become a target of cyber-attacks. Amongst many kinds of organization, the healthcare industry is one of the major targets that becomes a victim of cyber-attacks each year [15]. Followed by the digitalization of hospital operations and information systems, the amount of cyber-attack and variations rise as the technology developed. These incidents variant from breaches in personal health information to the larger size of attacks which can potentially halt hospital operations for a period. Halted in operation surely cause damage in various kinds. It may cost the hospital more than a million, or even cost individuals’ life because of the incident for the worst.

One of the major issues common amongst the healthcare industry is interoperability between each unit of the healthcare system. Especially, the interoperability between different organizations. Lack of interoperability prevents many opportunities for healthcare service quality improvement. The patient may need to take extra repetitive care procedures when visiting a new hospital. Mistakes in communication between different physicians can cause misdiagnosis. So, there are many demands from the patient side that want their health journey to be connected and improve healthcare service quality. However, interoperability is an extremely difficult issue for every single organization to solve. The foundation of healthcare informatics was developed separately by each organization. Each system has its own design and method to handle health information. That means there still have an open issue on how to solve interoperability in the field of healthcare. [7,9–11]

There are many kinds of incidents targeting healthcare industry. In recent years, one of major incidents found throughout the industry is hospital data breach. Data breach often appeared in a form that hospital data got compromised by hacker unnoticed by hospital employees. The compromised data can be valuable in dark market as it can be further used for various kind of more advanced attacks like identity theft, blackmailing, or social engineering, due to these data mostly included patients’ personal information and their health condition. This kind of incident can potentially cost hospital ‘a trust’ from their customer if they showed a poor quality of incident mitigation, as individuals’ safety and privacy are being put on the stake. Also, there are the case that not just gain unauthorized access to patient’s private data but, take over the data or even wipe all important data out of existence. ‘Ransomware’ and ‘Wipeware’ are the main cause of these threats. Ransomware take over an ownership over data away from hospital system and encrypt all the data which often take an important roles on hospital operation. At the same time, Wipeware will delete all the data from the victim machine. This mostly cause great disruption on hospital operation as consequence. Incidents that showed up in recent years seem to target healthcare organization more frequently, as the industry still have poor cybersecurity practices [16]. Many incidents [16–18] showed that social engineering launched on healthcare employees are on risen. The threat have potential to seamlessly blend into hospital workflow and made it hard to be noticed. However, follow these incidents, many stakeholders in healthcare domain start to implement cyber-security to their organization infrastructure.

At the foundation, each organization must start with educating their employees on cyber-security awareness to reduce risk of cyber-incident that may cause by human error or human vulnerabilities. Next, define organization policy and management plan that help prepare against cyber-incident. When employees and management level of organization have prepared cyber-security, then, the organization will focus on cyber security of technology layer. There are various kind of tools and technology that was invented to mitigate cyber-incidents. Some may have been made to prevent exploitation of existing technology while some may have been made to directly deal with known and upcoming threats.

One of many concepts invented to mitigate these threats is decentralization of data. The concept of decentralization was made to mitigate most incident and threat that involve single-point of failure vulnerability. For the case of healthcare industry where loss of patient’s data can cause many major damages to the affected organization and their patient, decentralization of data can help reduce damage caused by the case. There is more than one benefit that healthcare document data can gain from decentralization. Decentralization allows patient’s data that scattered across healthcare domain in different organization to link to each other. As healthcare document data can scattered across different organization within healthcare industry, it also increases a chance that its copies can survive cyber-incidents. Even in case that document in one organization got compromised, there is a chance that copies of compromised data also exist in other organization. The survived copies can make substitute for the original that got compromised. However, this only possible if there are the point that let every organization in the network known which document exist in which organization. This is where the concept of IHE Cross-Enterprise Document Sharing Profile fit in. Combined with Blockchain technology that make the Document Registry entry persist and immutable, this ensure that every organization in the network will always know whereabouts of document they need within the network while the entry itself cannot be tempered or deleted by any actor with ill intention.

This work will introduce another way to allow health document sharing between healthcare organizations with increased protection against cyber-threats, by using combination of Blockchain and IHE Cross-Enterprise Document Sharing (XDS.b) Profile.

**1.2 Problem statement**

To allow sharing of healthcare documents between different healthcare organizations which require maintenance of its confidentiality while mitigate emerging cyber-threats on the healthcare domain that tamper with integrity and availability of data, there needs the health document exchange medium that has distributed, decentralized, persistent, confidential, and immutable availability characteristics.

**1.3 Objective**

1.3.1 Design and implement Document Registry Blockchain that follows the requirement for document registry defined in the XDS.b integration profile from IHE.

1.3.2 Design and implement Blockchain smart contract that gives the main function to Document Registry Blockchain as healthcare document registry.  
 1.3.3 Design and implement Blockchain smart contract that gives additional function to record healthcare document exchange between participate node.  
 1.3.4 Deploy and evaluate the functionality of Document Registry Blockchain.

**1.4 Scope of project**

1.4.1 Design and implementation of Document Registry Blockchain that followed requirements defined in XDS.b integration profile from IHE.  
 1.4.2 Design and implementation of Blockchain smart contract within Document Registry Blockchain that gives the main function as healthcare document registry and additional function as healthcare document exchange history record.

# 

**CHAPTER II**

**LITERATURE REVIEWS**

**2.1 Integrating the Healthcare Enterprise (IHE)**

Integrating Healthcare Enterprise initiative (IHE) is an initiative founded by healthcare professionals and industry with the objective to improve the way computer systems in healthcare share their information. IHE promotes the coordinated use of established standards such as DICOM and HL7 to address specific clinical needs in support of optimal patient care. Systems developed in accordance with IHE communicate with one another better, are easier to implement and enable care providers to use information more effectively. This helps enables accessibility to health information that is usable whenever and wherever needed. The initiative is responsible for providing specifications, tools, and services for interoperability. They also engage clinicians, health authorities, industry, and users to develop, test, and implement standards-based solutions to vital health information needs. [19] The initiative has a purpose to provide a convenient and reliable way of specifying a level of compliance to standards sufficient to achieve truly efficient interoperability.

**2.1.1 IHE Process**

Figure 2‑1 showing the IHE Process where the initiative brings together users and developers of healthcare information technology in an annually recurring four-step process. The process started with clinical and technical experts define critical use cases for information sharing then followed by technical experts create detailed specifications for communication among systems to address these use cases. They also selecting and optimizing established standards during this step. After that, the industry implements these specifications which would be called “IHE Profile” into their healthcare information technology system. The initiative then tests these implemented systems at carefully planned and supervised events called “Connectathons” to ensure that the resulting IHE Profiles will provide benefit for the implementer and make them compatible with the unique environment of the healthcare industry. The initiative committees follow the four-step annual process to address interoperability in a variety of clinical domains including Cardiology, Dental, Devices, Endoscopy, Eye Care, Information Technology Infrastructure, Pathology and Laboratory Medicine,   
Patient Care Coordination, Pharmacy, Quality, Research, and Public Health,   
Radiation Oncology, Radiology, Surgery.

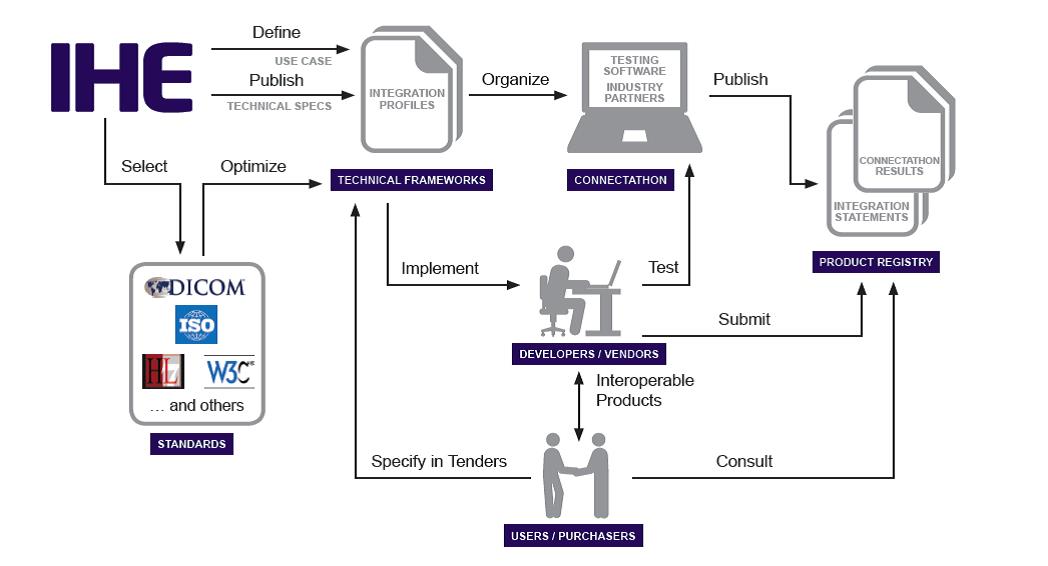


Figure 2‑1 IHE Process to create guideline for implementation of health information technology [20]

**2.1.2 IHE Integration Profiles**

IHE Integration Profiles or IHE Profile are products of the IHE Process which provide a standards-based framework for sharing information within care sites and across networks. They address critical interoperability issues related to information access for care providers and patients, clinical workflow, security, administration, and information infrastructure. IHE Profile was purposely designed to offers a clear implementation path for IT developers to develop and implement IT systems for a healthcare organization that meets the need and compatible with the environment of the healthcare industry while also aiding them in dealing with various kinds of communication standards existing within the healthcare IT domain. These profiles organize and leverage the integration capabilities that can be achieved by the coordinated implementation of communication standards, such as DICOM, HL7 W3C, and security standards. They provide precise definitions of how standards can be implemented to meet specific clinical needs. [21] Each profile specifically defines the actors, transactions, and information content required to address the clinical use case by referencing appropriate standards. IHE Profiles that have undergone IHE Process, sufficient testing, and deployment in real-world care settings and have reached final text (approved) status, will be published in specification documents called "IHE Technical Frameworks" (IHE TF). There is one Technical Framework per IHE clinical domain, with each framework comprised of multiple volumes. The Technical Frameworks provide detailed explanations for each IHE Profile specified by their interoperability issues and dependencies among the Integration Profile.

**2.1.3 IHE Information Technology Infrastructure Technical Framework**

IHE IT Infrastructure Technical Framework (ITI TF) defines specific implementations of established standards to achieve integration goals that promote appropriate sharing of medical information to support optimal patient care. It is expanded annually, after a period of public review, and maintained regularly through the identification and correction of errata. The framework identifies a subset of the functional components of the healthcare enterprise, called IHE actors, and specifies their interactions in terms of a set of coordinated, standards-based transactions. It describes this body of transactions in progressively greater depth. The framework divided into four volumes. The first volume describes concept detail of IHE ITI Integration Profiles. The second volume divided into four sub-volumes; a, b, c, and x which describe concept detail of all transactions present in the framework. The third volume provide further explanation into the specifications of cross-transaction and content used in Document Sharing Profiles. The fourth volume provide additional national extensions related to the framework.

**2.1.4 Cross-Enterprise Document Sharing Set-b (XDS.b) Profile**

The Cross-Enterprise Document Sharing Set-b (XDS.b) IHE Integration Profile facilitates the registration, distribution and access across health enterprises of patient electronic health records. [22] The profile is focused on providing a standards-based specification for managing the sharing of documents between any healthcare enterprises, ranging from a private physician office to a clinic to an acute care in-patient facility. XDS is generic term to reference all XDS profiles which are Cross-Enterprise Document Sharing Profiles. XDS.a and XDS.b are implementation profiles that describe technically how the implementation will be done. XDS-I is an XDS implementation specifically for medical imaging. [23] In IHE IT Infrastructure Technical Framework Vol.1 latest published in 2018 declared that term XDS within the ITI Technical Framework refers generically to any flavor of XDS, currently only XDS.b. [22] The main goal of XDS.b profile is to allow XDS Affinity Domain members to share health document via XDS Document Registry. That mean, its process mainly about make metadata of document within XDS Document Repository available on XDS Document Registry entry. This allow any XDS Document Consumer to visit XDS Document Registry and seek for the document they need, before retrieve it from the XDS Document Repository that the document belong to.

The process overview of Cross-Enterprise Document Sharing (XDS.b) profile is described in Figure 2‑2. The figure also showed sequence of process along with involving XDS actors and XDS transaction format. At the beginning, each health document will be created from its sources along with its metadata attributes. These sources will be called ‘XDS Document Source actor’ which can be any machine involved in healthcare service. For example, CT scanner, laptop in each physician office, or central computer in medical lab. Next, these created documents along with its metadata will be sent to data storage which act as document repository. These repositories will be called ‘XDS Document Repository actor’ which usually be some kind of computer or server that was assigned to keep medical document available for use. According to XDS.b profile, XDS Document Source will send document metadata in the format of Provide and Register Document Set-b (ITI-41) format. In some case, XDS Document Source and XDS Document Repository may integrated together. This made it called ‘XDS Integrated Document Source Repository actor’. The XDS Integrated Document Source Repository function the same way as XDS Document Source and XDS Document Repository will do but, combined together.

After the document and its metadata was sent to XDS Document Repository, the repository will index and make the document available for usage. At the same time, XDS Document Repository register metadata along with identifier and locator of the repository itself to local document registry. The message transaction in this process will follow format of Register Document Set-b (ITI-42). The document registry will be called ‘XDS Document Registry actor’. XDS Document Registry is software or machine that keep all document metadata and its corresponding repository from all connected repositories available for discovery. Commonly, XDS Document Registry should be database that keep document metadata from all connected repositories available for discovery through database query. However, there are no restriction from XDS.b profile for method to keep these data and how to discover each document metadata using specified document metadata attributes. There are just requirement that require XDS Document Registry to be able to accept value of specified document attributes from XDS Document Consumer and return the matched document to the consumer.

In XDS.b profile, ‘XDS Document Consumer actor’ can be any kind of software or machine that allow user like healthcare employees to access health document or medical document they need. There are no restriction in XDS.b profile that specified XDS Document Consumer actor to be different software or machine from other actors. XDS Document Consumer actor will just require user to specify value of known document metadata attributes which will allow XDS Document Repository to search for matching document metadata in its database. After received document attributes value from its user, XDS Document Consumer actor will send the specified attributes to XDS Document Registry. This message transaction will follow format of Registry Stored Query (ITI-18). Then, XDS Document Registry process received attributes by search for matching document metadata and return full document metadata which it found to XDS Document Consumer. XDS Document Consumer actor show founded result to its user. The user pick the right document they need and issue to XDS Document Repository corresponding to the document for document retrieval via XDS Document Consumer actor. XDS Document Consumer will send document retrieval request transaction in the format of Retrieve Document Set-b (ITI-43). After XDS Document Repository received document retrieval request from XDS Document Consumer, the repository will seek for the specified document and return the document to XDS Document Consumer. XDS Document Consumer actor will make the retrieved document available for user to use.

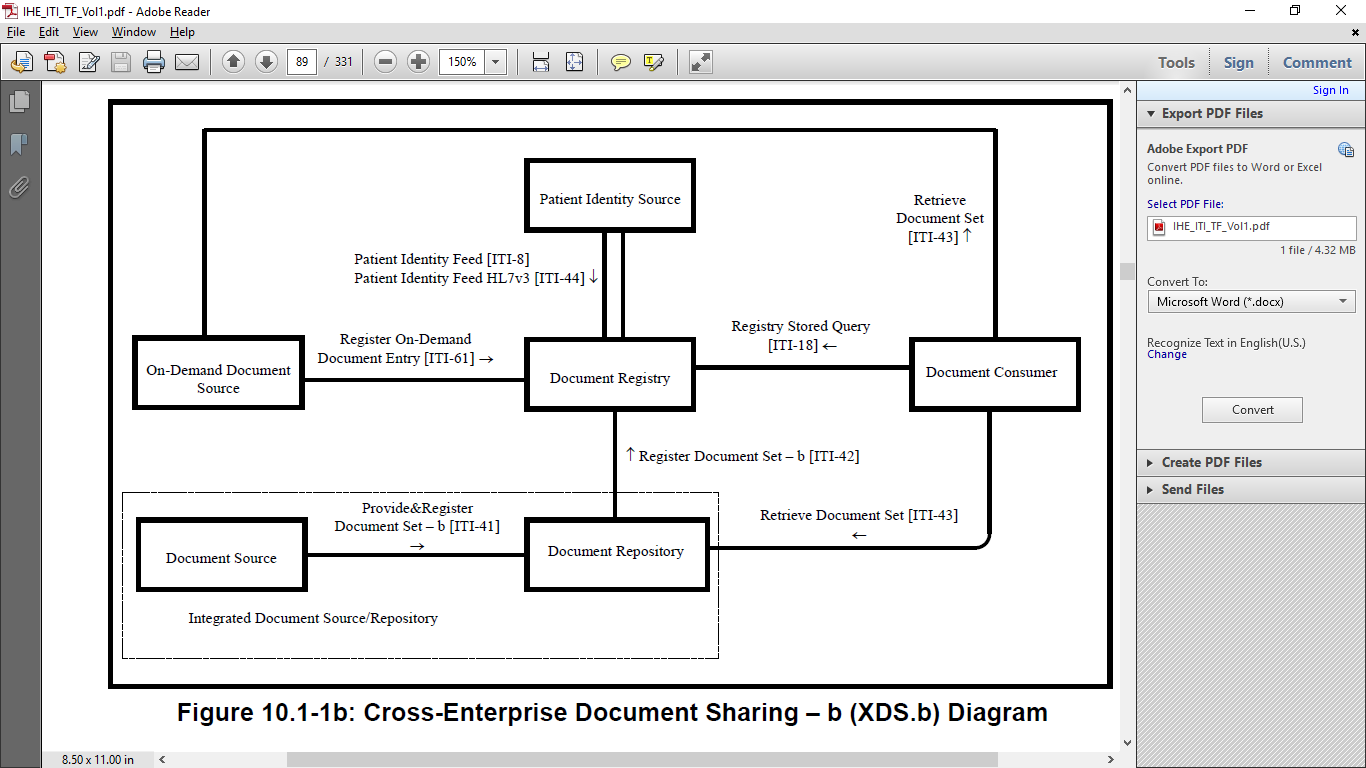


Figure 2‑2 Cross-Enterprise Document Sharing - b Diagram [22]

**2.1.5 XDS Transaction Format Types**

In XDS.b profile, all messaging transaction will be in the form of XML format with schema depend on each types of transaction. Types of XDS transaction format vary upon involving actors and its purpose.

2.1.5.1 Provide and Register Document Set – b (ITI-41)

Provide and Register Doccument Set – b (ITI-41) transaction format define XML schema for message that send metadata of document from XDS Document Source actor to XDS Document Repository actor for store into document repository. This type of transaction mainly require XDS Document Source to include all available metadata attributes of created document for other XDS actor. XDS Document Repository actor will need to acknowledge to XDS Document Source if it successfully received document and its metadata.

2.1.5.2 Register Document Set – b (ITI-42)

Register Document Set – b (ITI-42) define XML schema for message that send metadata of available document in repository from XDS Document Repository actor to XDS Document Registry actor to register the document into document registry entry. Main purpose of this type of transaction is to pass document metadata stored in repository to XDS Document Registry actor addition with attributes about the repository. XDS Document Registry actor will need to respond back to XDS Document Repository actor when received the transaction and register it to document registry entry.

2.1.5.3 Registry Stored Query (ITI-18)

Register Stored Query (ITI-18) is general XML schema format that used by one actor to query for data from other actor in entire IHE IT Infrastructure Framework. In this work, the transaction will be used by XDS Document Consumer actor to request for document metadata it seek from XDS Document Registry actor. Any document metadata attributes known by XDS Document Consumer will be included in the transaction. XDS Document Registry will use specified metadata attributes to search for matching document metadata inside document registry entry. XDS Document Registry will need to respond to XDS Document Consumer actor that it received the request. XDS Document Registry also need to return search result to XDS Document Consumer.

2.1.5.4 Retrieve Document Set (ITI-43)

Retrieve Document Set (ITI-43) define XML schema for XDS Document Consumer to request document retrieval from XDS Document Repository. Different to other transactions involved in XDS.b profile, Retrieve Document Set transaction only contain few essential attributes to allow retrieval of document from document repository. XDS Document Repository will need to acknowledge to XDS Document Consumer when received the transaction before return the requested document.

**2.1.6 Transaction Object Type and Metadata Attributes**

In each transaction, there are set of metadata attributes that represent the document. These metadata attributes are categorized to three sections. SubmissionSet (Table *1*) represent information associated with submission of document since it was created by the source. Folder (Table *2*) represent group that the document belongs to. DocumentEntry (Table *3*) represent the document itself.

2.1.6.1 SubmissionSet

Table 1 SubmissionSet Metadata Attributes

|  |  |
| --- | --- |
| **SubmissionSet Metadata Attributes** | **Description** |
| author | The humans and/or machines that authored the SubmissionSet. This attribute contains the sub-attributes: authorInstitution, authorPerson, authorRole, authorSpecialty, authorTelecommunication. |
| availabilityStatus | The lifecycle status of the SubmissionSet. |
| comments | Comments associated with the SubmissionSet. |
| contentTypeCode | The code specifying the type of clinical activity that resulted in placing the associated content in the SubmissionSet. |
| entryUUID | A globally unique identifier used to manage the entry. |
| homeCommunityId | A globally unique identifier for a community. |
| intendedRecipient | The organizations or persons for whom the SubmissionSet is intended. |
| limitedMetadata | A flag that the associated SubmissionSet was created using the less rigorous metadata requirements as defined for the Metadata-Limited Document Source. |
| patientId | The patientId represents the primary subject of care of the SubmissionSet. |
| sourceId | Identifier of the entity that contributed the SubmissionSet. |
| submissionTime | Point in time at the creating entity when the SubmissionSet was created |
| title | The title of the SubmissionSet. |
| uniqueId | Globally unique identifier for the SubmissionSet assigned by the creating entity. |

2.2.6.2 Folder

Table 2 Folder Metadata Attributes

|  |  |
| --- | --- |
| **Folder Metadata Attributes** | **Description** |
| availabilityStatus | The lifecycle status of the Folder. |
| codeList | The set of codes specifying the type of clinical activities that resulted in placing DocumentEntry objects in the Folder. |
| comments | Comments associated with the Folder. |
| entryUUID | A globally unique identifier used to manage the entry. |
| homeCommunityId | A globally unique identifier for a community. |
| lastUpdateTime | Most recent point in time that the Folder has been modified. |
| limitedMetadata | A flag that the associated Folder was created using the less rigorous metadata requirements as defined for the Metadata-Limited Document Source. |
| patientId | The patientId represents the primary subject of care of the Folder. |
| title | The title of the Folder |
| uniqueId | Globally unique identifier for the Folder. |

2.2.6.3 DocumentEntry

Table 3 DocumentEntry Metadata Attributes

|  |  |
| --- | --- |
| **DocumentEntry Metadata Attributes** | **Description** |
| author | The humans and/or machines that authored the document. This attribute contains the sub-attributes: authorInstitution, authorPerson, authorRole, authorSpecialty and authorTelecommunication. |
| availabilityStatus | The lifecycle status of the DocumentEntry |
| classCode | The code specifying the high-level use classification of the document type (e.g., Report, Summary, Images, Treatment Plan, Patient Preferences, Workflow). |
| comment | Comments associated with the document. |
| confidentialityCode | The code specifying the level of confidentiality of the documented. |
| creationTime | The time the author created the document. |
| entryUUID | A globally unique identifier used to manage the entry. |
| eventCodeList | This list of codes represents the main clinical acts, such as a colonoscopy or an appendectomy, being documented. |
| formatCode | The code specifying the detailed technical format of the document. |
| hash | The hash of the contents of the document. |
| healthcareFacility TypeCode | This code represents the type of organizational setting of the clinical encounter during which the documented act occurred. |
| homeCommunityId | A globally unique identifier for a community. |
| languageCode | Specifies the human language of character data in a document. |
| legalAuthenticator | Represents a participant within an authorInstitution who has legally authenticated or attested the document. |
| limitedMetadata | Indicates whether the DocumentEntry was created using the less rigorous requirements of metadata as defined for the Metadata-Limited Document Source. |
| mimeType | MIME type of the document. |
| objectType | The type of DocumentEntry (e.g., On-Demand DocumentEntry). |
| patientId | The patientId represents the subject of care of the document. |
| practiceSettingCode | The code specifying the clinical specialty where the act that resulted in the document was performed (e.g., Family Practice, Laboratory, Radiology). |
| referenceIdList | A list of Identifiers related to the document |
| repositoryUniqueId | The globally unique identifier of the repository where the document can be accessed. |
| serviceStartTime | The start time of the service being documented. |
| serviceStopTime | The stop time of the service being documented. |
| size | Size in bytes of the document. |
| sourcePatientId | The sourcePatientId represents the subject of care’s medical record identifier (e.g., Patient Id) in the local patient identifier domain of the creating entity. |
| sourcePatientInfo | This attribute contains demographic information of the source patient to whose medical record this document belongs. |
| title | The title of the document. |
| typeCode | The code specifying the precise type of document from the user perspective (e.g., LOINC code). |
| uniqueId | Globally unique identifier assigned to the document by its creator. |
| URI | The URI for the document. |

**2.2 Blockchain Technology**

**2.2.1 Definition of Blockchain**

Blockchain is a list of records, or “blocks”, that are linked to one another and cryptographically secured [24]. Blockchain is a technology that allows data to be stored and exchanged on a peer-to-peer basis. Structurally, Blockchain data can be consulted, shared and secured thanks to consensus-based algorithms [25]. Blockchain is a sequence of blocks, which holds a complete list of transaction records like conventional public ledger [26]. Participants in a Blockchain network have records of every transaction and these records are stored locally on the computers of all participants in that Blockchain network. Any kind of regime or protocol change to a Blockchain network requires consensus between the users of the network. In 2008, the Blockchain idea was combined with several other technologies and computing concepts to create modern cryptocurrencies which is electronic cash protected through cryptographic mechanisms instead of a central repository or authority.

This technology became widely known in 2009 with the launch of the Bitcoin network as the first widely known Blockchain network with its purpose as cryptocurrency, followed by multi-purpose Blockchains like Ethereum and many other platforms with their own application. For cryptocurrency Blockchain like Bitcoin, the transfer of digital information that represents electronic cash takes place in a distributed system. Bitcoin users can digitally sign and transfer their rights to that information to another user and the Bitcoin Blockchain records this transaction into Blockchain's distributed ledger, make it available for public verification. Blockchain was designed to defy the concept of having a single centralized system as the host of the network while subsequently allow the concept of decentralization to take the place by having many members of the network to equally maintain the replicated distributed ledger. Combined with the cryptographically hashed "Block" and "Chain" concept, makes the Blockchain resilient to any attempts to alter information recorded in the distributed ledger. With the contribution of the Blockchain developer community, the technology now available for a variety of applications and being researched for a variety of further usage in many industries. [27]

According to the document “Blockchain Technology Overview” [27] which published by National Institute of Standards and Technology from U.S. Department of Commerce, Blockchain can be informally define as: A distributed digital ledgers of cryptographically signed transactions that are grouped into blocks. Each block is cryptographically linked to the previous one (making it tamper evident) after validation and undergoing a consensus decision. As new blocks are added, older blocks become more difficult to modify (creating tamper resistance). New blocks are replicated across copies of the ledger within the network, and any conflicts are resolved automatically using established rules.

**2.2.2 Benefit of Blockchain**

Blockchain tampers evident and tamper-resistant digital ledgers implemented in a distributed fashion and usually without a central authority. At their basic level, they enable a community of users to record transactions in a shared ledger within that community, such that under normal operation of the Blockchain network no transactions can be changed once published [27].

**2.3 Ethereum** **and Smart-Contract**

Ethereum is one of the well-known open-source Blockchain platforms. The platform initially invented by the developer named Vitalik Buterin and further develop and maintain by the Ethereum community [28]. The main approach of Ethereum Blockchain is about the use of Blockchain technology for applications other than cryptocurrency. The platform was the first to propose the concept of a ‘smart contract’ that enables programming over Blockchain technology.

**2.3.1 Blockchain Characteristics**

Key characteristics of the Blockchain can be vary depend on its setup and environment of usage. According to many sources, key characteristics of the Blockchain may be summarized as followed:

2.3.1.1 Decentralization

Decentralization is the foundation of Blockchain technology as response to problem of centralized system. In centralized system, especially centralized database, there is a chance that the database got compromised by hacker. Other than rely on backup data, there are very few options to deal with the incident. This makes the compromised database become single point of failure which prevent follower system to operate. Decentralization of data was proposed to scatter the chance of single database from getting compromise. This makes decentralized database network have more resistant against incident threatening centralized data. Even hit by incident that aim to compromise the data. If at least half of decentralized network survived the incident, the data survive the attack.

2.3.1.2 Immutable

With utilization of cryptographically hashed chain combined with decentralized network, the Blockchain technology ensure that any data published on Blockchain cannot be deleted or modified. If there are any modification made to content of published data, it will cause change on the hash chain and detected the network. Any action that causes change to hash chain will be negate by majority of the network. This mean if anyone want to temper with published data on Blockchain, they will need to compromise the entire network at once. Any survived node has chance to notify the abnormal to the entire network.

2.3.1.3 Transparency

As the foundation of Blockchain is to have all participant nodes have the exact same copy of Blockchain ledger, it passively gives transparency to published data. It is impossible for anyone to secretly hide something inside Blockchain without let other participants in the Blockchain network know.

2.3.1.4 Distributed

Blockchain have distributed characteristic by design. All nodes will have the same Blockchain ledger. Any content published to Blockchain ledger are passively distributed to all Blockchain node. With consensus algorithm, it requires that the publishing content either sent to all nodes before accepting to publish or being accepted then send to all node, to complete consensus. So, Blockchain ensure that any data published to the chain are distributed to all connected nodes.

2.3.1.5 Trusted

In public network where anyone can participate or in permissioned network where participants are not completely trust each other, trust is the main factor that define usability of decentralized network. Along with Blockchain technology, consensus solve the issue about trust by eliminate the chance of any single node participate in Blockchain to have absolute control over publishing data when certain condition is met. It can rely either on randomness or specially designed algorithm depend on each consensus method. When none of any single node can have absolute control over publishing data on the Blockchain, made it extreme difficult for someone to temper with target data. Many consensus methods ensure that it will much more expensive for anyone to attempt on tempering with publishing data when compared to benefit they can get. This passively establish trust between all participant nodes.

**2.3.2 Blockchain Types**

When considering the scope of participants who can participate in a specific Blockchain network, Blockchain can be categorized into 3 types of Blockchain network.

2.3.2.1 Public Chain

Public Blockchain is the type that allows anyone to participate in the network either participating as client/user node or miner/validator node. This type of Blockchain mostly has no specific rule, policy, or agreement for participants to enter the network. The type suited best with the network environment that confidentiality from the public is not required for information recorded in the Blockchain ledger and encourages public verification for its transparency.

2.3.2.2 Private Chain

Private Blockchain is the type that allows only a limited number of members to participate. This type was invented to be more compatible with the environment that participant nodes are members of a specific organization or community where the Blockchain ledger may record confidential information limited only to participants.

2.3.2.3 Permissioned Chain

Permissioned Blockchain is the type that allows only selected members of a specific community or affinity domain to participate, and it is also known as consortium Blockchain. Permissioned Chain is different from Private Chain in terms of scalability. As the private chain was limited for pre-selected members, the permission chain may further extend its member to a larger group of members via policy or agreement accepted by original participants. At the same time, the permission chain will not allow anyone to participate in the network as the Blockchain ledger may contain confidential information limited to the accepted group of participants.

**2.3.3 Blockchain Components**

2.3.3.1 Transaction and ‘Block’

Each of individual information represent change or cause of actions in information system are stored within Blockchain as “Transaction”. Several transactions being publish to Blockchain within the same time interval are put in the same “Block”. To form each single block, miner or validator need to hash transaction together. The resulting hash value represent integrity of each block. If there are any change apply to transaction in the block, it will cause hash value of the block to change. Format of block vary depend on each Blockchain platform and its use case. Some platforms may publish in a form of plaintext just to act as the source of truth for every participating node to look without constraint. Some platforms may bound transaction or block to unique address to extend variation in accessibility. Some platforms may encrypt block to maintain confidentiality of data. Transaction and Block are the key component which determine purpose and application of Blockchain.

2.3.3.2 Cryptographically hashed ‘Chain’

Other than the concept of “Block”, The Blockchain concept also introduced the concept of “Chain. As integrity of each Block represent by its hash value, integrity of entire Blockchain represent by all hash value of all Block within “Chain”. The foundation of “Chain” concept is by chaining hash value of all blocks together. This can be done by include hash value of block formed in previous time interval into the current block to generate its hash value. Any changes made to any one single block will alter hash value of the entire chain that come after. This makes it harder to alter data that published within Blockchain. It requires anyone who want to alter the data to apply change to all block that come after the target block until the current one to make the change valid. Combined with decentralization characteristic of Blockchain network, this makes data exist in Blockchain nearly impossible to alter.

2.3.3.3 Distributed network of participate ‘Node’

Any machine that participates in the Blockchain network is called “Node”. Nodes represent the population of each Blockchain network. Each node keeps the same copy of data in Blockchain which is the Blockchain ledger and always tends to in sync with each other via Blockchain-specific protocol depend on each platform. In the case of Ethereum, the platform utilizing devp2p protocol which allows each Ethereum node to connect with others at a peer-to-peer level [29]. If there are any differences in data between nodes, the version of data being held by a minority of participating nodes will be clarified as false before rejected by the entire network and replaced by the right version accepted by the network. In each Blockchain network, some nodes may participate as "miners" or be elected as "validators" of the network at each different time interval. Miner/Validator nodes have a duty to perform the task which maintaining the consensus of the network. The Blockchain can be alive only if they are at least one participating node to maintain it, while the strength of its characteristics depends on the number of participating nodes. In most cases, more participating nodes mean stronger Blockchain.

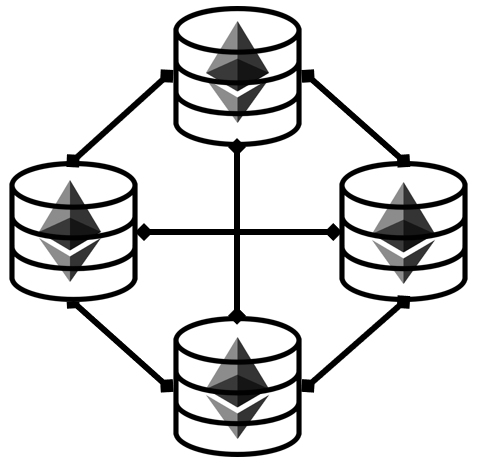


Figure 2‑3 Blockchain network formed from the participation of Blockchain nodes

2.3.3.4 Consensus

Each Blockchain network must have its own consensus within the network. Specifically, it is not just the consensus between the member but the built-in consensus mechanism within the communication protocol connects each member in the network. The consensus mechanism acts a vital role to ensure the integrity of the Blockchain ledger by ensuring that everyone in the network is holding the same copy of the Blockchain ledger and no one has full control over data adding to the Blockchain ledger or have the right to select which version of the Blockchain ledger for the network to maintain. There are many variations of consensus invented since the beginning of Blockchain technology. Each has its own method of investing resources to achieve complete consensus. The existing consensus concept at the time can be divided into 3 major types based on the participation of Blockchain members who declared to act as maintainers of the Blockchain ledger.

2.3.3.4.1 Competition-based consensus

The type was the first to introduced along with the beginning of Bitcoin Blockchain which called "Proof of Work" (PoW). The competition-based consensus like PoW requires Blockchain maintainer node (which was called "miner") to invest computational resources on competing with other miners to solve the specific mathematical puzzle which only computational power can effectively solve. The first miner who can solve the puzzle will have the right to add the adding Block to the Blockchain ledger and receive a reward declared by the network which would compensate the investment. In the case of PoW, a miner must randomly generate a correct "nonce" number that when hashing together with the hash value of the adding Block will result in a hash value with a digit of '0' beginning specified by the network (such as 0x0000000abcd). This kind of puzzle ensures that the chance where malicious actors want to attempt malicious activity on the specific transactions on certain Blocks is at least possible. Combined with the increase in the number of miners entering the competition given that chance becomes nearly impossible to achieve. This consecutively ensures the integrity of the Blockchain ledger and gives transparency to Blockchain. The scenario of competition-based consensus suited best to Public Blockchain where anyone can participate in Blockchain. The more miner entering the competition means the more reliability and transparency for the Blockchain. Additionally, the reward-based nature of the concept can even further synergize the Blockchain to have more miners participate in the network. However, due to the competition will have major of computational power invested in achieving consensus, that means the environment where computational power is limited and precious to its member will not be compatible with this type of consensus.

2.3.3.4.2 Randomness-based consensus

The type was originally introduced as an alternative to a competition-based consensus like PoW and to address the problem where major computational power will be wasted in achieving consensus. The concept proposes utilizing randomness to aid in the selection of the Blockchain maintainer at a certain time. Widely known consensus mechanisms that can be categorized to this type are "Proof of Stake" (PoS) and "Proof of Authority" (PoA) which using pseudo-random algorithm combined with additional factor to determine for the node that has the right to add Block to the Blockchain ledger at a certain time (which would be called "validator). Both PoS and PoA require validator candidate to place "the bet" on "the stake" which would mostly be cryptocurrency circulating in the network. The one with a higher bet put on a stake will have a higher chance to be selected as the validator of the time. However, there is still a chance that the one with a lower bet can be selected as the validator instead. The one that has been selected as the validator will gain all the bet placed on the stake. That means each candidate needs to take an equal risk to gain and loss their available bet. This consecutively results as distributed right amongst the network similar sense to what achieved in competition-based consensus.

As this type of consensus act as an alternative to competition-based consensus, that means the environment best suited with this type of consensus is where its member cannot effort to lose computational power in competition-based consensus altogether.

2.3.3.4.3 Majority-based consensus

This type of consensus was also introduced as another alternative to a competition-based consensus like PoW. The original concept of this consensus type was originated from the "Practical-Byzantine False Tolerance" (PBFT) method. The method was invented for the traditional logic systems to determine for decision the system would take in the assumption that the majority of its members are on the "good side" and will take responsibility to help the system achieve the best decision. Implement to Blockchain, the consensus mechanism requires all participate node to act in a similar fashion to the validator. The network will only accept the adding Block to the Blockchain ledger when the adding version is the similar version in a majority of the network. This means it requires the adding Block to be the version that 2/3 of all member nodes propose to add to the Blockchain ledger. This eliminates the chance where malicious actor which assumed to be the minority of the network to attempt malicious activity on the adding Block. However, due to the majority-based nature of the type, it is only compatible with Blockchain types with only known members including Private and Permissioned Blockchain.

**2.3.4 Smart Contract**

The concept of smart contract was initially proposed by Ethereum [28, 30]. Now the word ‘smart contract’ become common word to describe feature that allow developer to design the content that publish to Blockchain and its computational behavior. In Ethereum, smart contract code written with Solidity programming language. Smart contracts define what behavior the contract will do when open/view by user. Smart contracts rely on Ethereum Virtual-Machine (EVM) which allow host machine of Ethereum client to be able to execute smart contract Solidity code. EVM was designed to allow portability of Ethereum platform and always packed with Ethereum client. Now there are many interface tools developed by Ethereum community that allow Ethereum client to work with major programming languages. This further extend usage of smart contract to infinite possibilities.

2.3.4.1 Solidity [30]

Solidity is Javascript-like programming language that specifically design to use with Ethereum smart contract. The main purpose of the programming language is to act as the middle between human-understandable language and computer language. It reduces difficulty for developer to design behavior of their smart contract on Ethereum Blockchain. The language is update and maintain by Ethereum community.

2.3.4.2 Quorum [31, 32]

Quorum (or later renamed as GoQuorum) is an Ethereum-based distributed ledger protocol forked of go-ethereum enabled for transaction/contract privacy and a wider range of majority-based consensus mechanisms compatibility. The platform was initially developed by "JPMorgan" and further developed and maintained by the Quorum community. Quorum enables the usage of PBFT by inventing a PBFT-inspired consensus algorithm called Istanbul BFT which was adapted to be compatible with the Ethereum Blockchain environment. The platform also offers the “7-Nodes Example” [33] for developers to invent and test their Blockchain concept which is useful for the development with the limited computational resources available.

**2.4 Related Work**

There are many research proposing about decentralize healthcare information with Blockchain technology. The goal of decentralization and implementation of each work have many variants. These are several works that proposed interesting idea and concept about implement healthcare informatics system based on Blockchain technology.

**2.4.1 A Blockchain-Based Approach to Health Information Exchange Networks** [34]

The work proposed about using Blockchain like central hub for health information exchange. The main goal of this Blockchain concept is to connect all bread and crumb of patient health information together by allow participate node to discover health information data they seek and its location within Blockchain ledger. Increase interoperability in health information exchange. Their main contribution is the concept that suggest use of FHIR health information exchange standard combine with Blockchain technology. Each transaction on Blockchain will contain FHIR locator of actual data along with its index which make each transaction available for search. Due to the limit of health information that it requires certain amount of confidentiality, this make it not really compatible with platform open to public like Blockchain. Store actual data somewhere else outside Blockchain and put its locator into Blockchain for use. With known secure index, this Blockchain help connect patient information that scattered across healthcare industry together. The work also gave suggestion about how health information Blockchain should look like and what it should have by common. There also other major contributions that proposed about using secure index for searching on encrypted data and ‘Proof of Interoperability’. This work suggests that if health information are kept within Blockchain in encrypted form, it should also contain secure index which will allow data search even the data is encrypted. This should reduce the difficulty of implementing health information with Blockchain. And other major concept proposed in this work is ‘Proof of Interoperability’. Based on Proof of Work consensus, the work suggest that computational resource should not be wasted unnecessarily. Instead of put computational resource to competition for consensus, it should be used to verify interoperability of participate health data instead. However, they didn’t proposed about how the consensus should work in detail. This work gave a good example of how Blockchain can have potential to solve issue that common in healthcare industry like interoperability. Additionally, they also proposed many concepts that can be a good foundation for using Blockchain technology with health information.

**2.4.2 A Case Study for Blockchain in Healthcare: “MedRec” prototype for electronic health records and medical research data** [35]

Main goal of MedRec is to provide Blockchain that act as a middle for health information exchange while allow Blockchain participants to gain benefit from participation. They chose Ethereum as Blockchain platform for the system. Ethereum provide smart contract and address based access for the work. This work assume that miner/validator nodes are health institution that have demand for large amount of health information data to use in their research. Miner/validator node will be rewarded with anonymized health data which can be used in research involve health data analysis. Additionally, MedRec proposed about allowing patient to have consent about usage on their data. Give more control over individual health data. The work also adopted cryptographic key scheme proposed by Zyskin et al. [36], to ensure that only authorized party can access patient health information published on Blockchain. Additional to these main contributions, they also gave suggestions about factor that should keep continuity of Blockchain and how Blockchain element provided by platform like Ethereum can be useful. One of interesting concept is about using Ethereum address as patient identifier. Due to all identity exist on Ethereum Blockchain are assigned with unique address, these unique addresses can reduce complexity in patient identifier management if designed properly. MedRec gave a good example of concept that needed to maintain continuity of Blockchain network by allow participant to gain benefit from participation in some way. At the same time, MedRec is another good example that using Blockchain technology to aid health information exchange issue. And the last, MedRec have shown flexibility of smart contract and how it can be useful when implement with healthcare information.

**2.4.3 Blockchain-Based Data Preservation System for Medical Data** [37]

This work proposed about using Blockchain to keep data that need to have confidentiality preserved. Regardless of what kind of data, this Blockchain allow user to design what data they want to keep in Blockchain. The chosen data will be encrypted before publishing into Blockchain. The goal of this Blockchain concept is to preserve medical data inside Blockchain away from any tempering attempt while keep it secret and always available for its owner. Instead of let data available to public, this work has demonstrated how Blockchain technology can be used in different approach like keeping medical data available to only authorized entity.

**2.4.4 Blockchain-based electronic healthcare record system for healthcare 4.0 applications** [38]

The work has gathered research proposing the Blockchain concept from 2016 to 2019 that would benefit the healthcare industry by enhancing the capabilities of electronic health records. The work has well explained the overall concept of implementing Blockchain technology to electronic health records developed over the years. They also proposed another approach of implementing Blockchain technology for electronic health records by using IBM Hyperledger fabric as a medium for health information exchange in a similar fashion with MedRec which prioritize efficiency in handling huge number of transactions in the meantime. The contribution in the work inspires and encourages the idea of enhancing the existing EHR system with Blockchain technology as it provides the characteristics of distribution and decentralization.

# 

**CHAPTER III  
PROPOSED METHOD**

This chapter explains the proposed method. The first section introduces the use case scenario of the proposing concept. The second section explains the main concept design of this work. The third section then explains the design of Blockchain infrastructure that would be compatible with implementation on the IHE XDS.b Profile. The fourth section explains the integration of Blockchain technology into the IHE XDS.b Profile. The fifth section further explaining the detail regarding the main function of our implementation. The last section shows the process flow of the XDS Document Registry Actor as an interface between the XDS system and the Blockchain ledger.

## 3.1 Use case scenario

Assume that the case started with Mr.Bob once got an illness and visited Hospital A. At the time, a physician at Hospital A diagnoses that Mr.Bob has influenza and allergy to some medication. Sometimes has passed, Mr.Bob got an illness again but design to visit Hospital B. A physician at hospital B then wants to know the history of health and drug allergy of Mr.Bob. Not known to a physician at Hospital B, the information needed is available at Hospital A. With the electronic health record system of both hospitals fully complied with the XDS Profile, the physician at Hospital B will just need known information of Mr.Bob to discover and gain access to the latest health document of Mr.Bob available at Hospital A using XDS Document Registry. However, if the organization or the machine maintaining the XDS Document Registry went down due to an unexpected incident, this will prevent the physician at hospital B to access Mr. Bob's health document at Hospital A. Additionally in the worst case, if the data of Mr.Bob's health document become permanently inaccessible, the health document sharing network would lose the opportunity to use the document for their operation forever even the XDS Document Registry become available. This is where the Blockchain integrated XDS Document Registry take the place. With Blockchain characteristics mentioned in section 2.3.1, the XDS Document Registry will always available and not depend on just a single organization or a machine to maintain its availability while providing the possibility for the data to survive the situation where it may be permanently lost.

## 3.2 Concept Design

As introduced in Chapter I, the unique nature of the healthcare environment that emphasizes confidentiality of data gave limits to implementing Blockchain technology into the industry. Patient data cannot be put directly into Blockchain as it will become persistent following Blockchain characteristics while increasing difficulty in ensuring data confidentiality when its replica is distributed all over the network [26, 27, 39]. So instead of risk confidentiality of healthcare data by publishing it directly into the Blockchain network, we propose using IHE XDS.b Document Registry Actor to act as a health document exchange medium for the Blockchain network. The profile is best compatible with Blockchain technology as decentralization will secure the availability of the health information exchange by eliminating the need for the organization that will act as the central hub for the exchange avoiding a single point of failure problem. At the same time, there is no longer a need to publish health documents directly into the distributed network, reduce the risk against the confidentiality of the data. Additionally, in this work, we further extend the usability of the profile by allowing the organization that has shared health documents from its source to also act as an additional data backup for the original by providing additional access points (URLs) for the document. That mean, even the source of health document become unavailable due to unpredictable circumstance like a cyber-incident (i.e., Ransomware threat), the network will still have a chance to access the compromised document via an alternative source available from a shared peer. This extends the benefit of the health document sharing network and encourages the growth of the health document sharing community even further indefinitely.

Following Figure *2‑2*, the XDS Document Registry actor who acts as a hub for health document exchange would normally host a database that allows XDS Document Consumer Actor to query for information of health Document they seek. The existing solution for the database is the utilization of SQL or non-SQL centralized database depend on each XDS Affinity network. In adaptation for this work, we propose replacing these centralized databases with a Blockchain ledger which innately distribute the registry amongst the network while also benefit from Blockchain characteristics. This consecutively transforms the XDS Document Registry Actor host by each XDS Affinity domain member into a Blockchain node. Each node will now serve as a decentralized XDS Document Registry Actor who will joint cooperatively keep, operate, and maintain the Blockchain ledger which now contains the entire health document registry entry for the network.

## 3.3 Blockchain Design

Considering the environment of the XDS Affinity domain network, the network is comprised of members who are hospital or health institution that entered the network intending to share their health document and access health document shared from other. Each member entering the network are expected to have been selected by the network and have an agreement with the network to voluntarily share health document from their XDS Document Repository Actor to the network while not exposing information shared within the network to the outside. Following Blockchain Types mentioned in 2.3.2, the network is not suitable with the Public Chain as they are not accepting anyone into the network without pre-selection and a proper agreement. At the same time, they are also not suitable with the Private Chain as the XDS Affinity domain network was designed for scalability and not fixed to any specific organization. That means the XDS Affinity domain network is best suitable with the Permissioned Chain type as it was designed for scalability welcoming more members to join the network over time under the condition that joining members are accepted by the network and have a proper data sharing agreement.

Meanwhile, following the Blockchain consensus mentioned in 2.3.3.4, the XDS Affinity domain members were not pre-determined to invest a high amount of computational resources for entering the network and were not expected to gain direct profit from participation in maintaining the network. That means the consensus mechanism with computational resources inefficiencies like competition-based and profit-dependence mechanism like randomness-based are not the choice of consensus for the network. Then the remaining majority-based consensus will be the most suitable choice available. The start point of majority-based consensus will be PBFT as the basis of the type. With the nature of the XDS Affinity domain network which pre-determined the joining member, this should prevent the chance for the bad actor to control the majority and secure Blockchain characteristics for the network consecutively.

## 3.4 Integrating Blockchain with XDS.b Profile

For the implementation, following Figure *2‑2*, we assume that the ITI-61 transaction is unessential for the current usage of the work. The patient identification was assumed to be already standardized amongst all members of the network, eliminates the need for the ITI-44 transaction from consideration. This left the XDS Document Registry Actor Blockchain to only interact with the remaining XDS Document Repository Actor via ITI-42 transaction and XDS Document Consumer Actor via ITI-18 transaction (as shown in Figure *3‑1*).

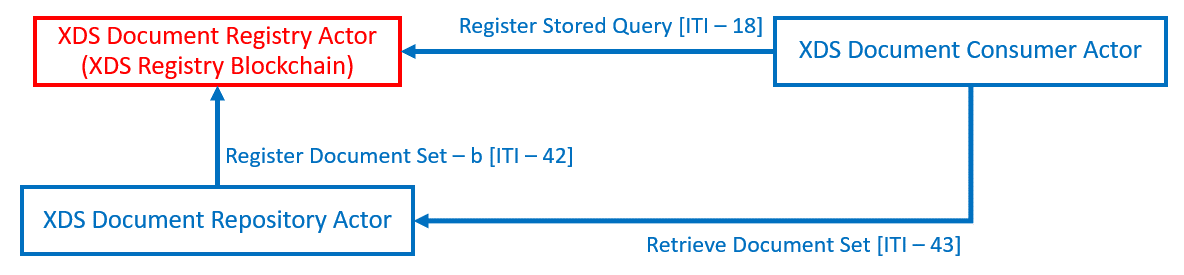


Figure 3‑1 XDS Profile within the scope of interest for this work

Following Section 2.1.4, all Blockchain nodes will receive ITI-42 transactions from their local XDS Document Repository Actor as normal XDS Document Registry Actor would do, before transitioning the transaction into Blockchain Smartcontract and publish into the Blockchain ledger. Likewise, the health document query via ITI-18 transactions from local XDS Document Consumer will be interpreted and interact with Smartcontract consecutively. Note that the XDS Document Consumer Actor will still be required to directly issue ITI-43 transaction to the XDS Document Repository Actor hosting the health document to retrieve it. The Smartcontract will act as a medium for each node to perform the task to add data, read data, and search for data within the Blockchain ledger (as shown in Figure *3‑2*) allow the Blockchain technology to effectively integrated into the XDS system.

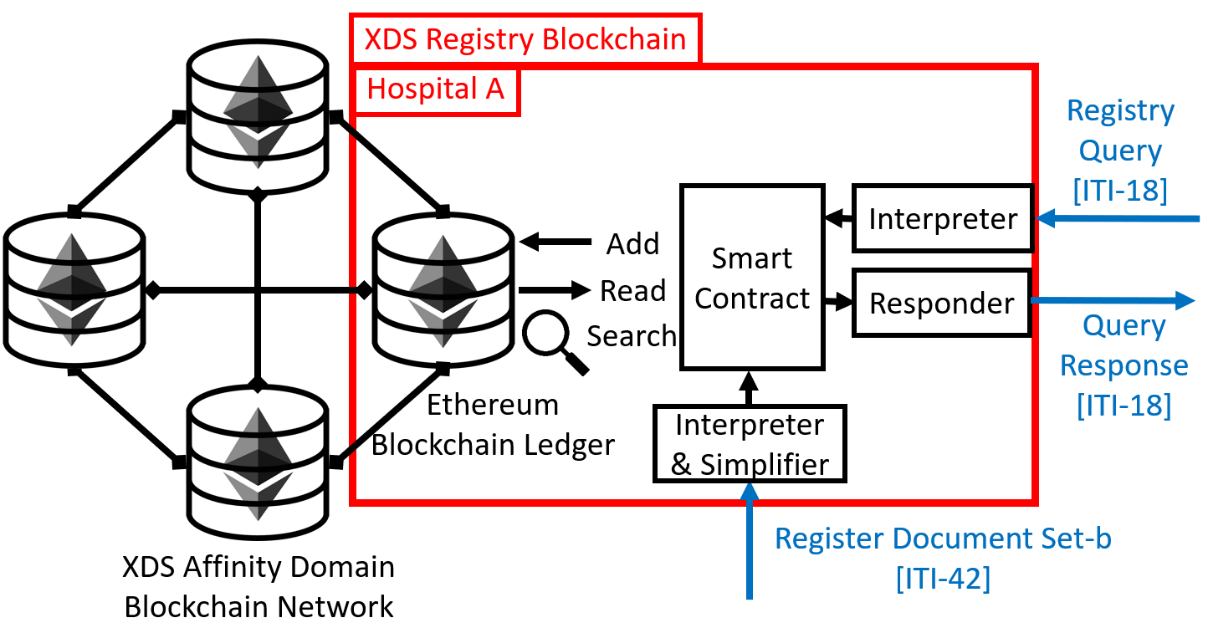


Figure 3‑2 Integrating Blockchain into XDS.b Profile

## 3.5 Design Functions

**3.5.1 Document Register**

This is where XDS Document Registry Actor registers health document metadata set within ITI-42 transaction received from XDS Document Repository into the Blockchain ledger. This function enables sharing of health documents to the XDS Affinity Domain Blockchain network as well as allowing the shared document to be registered as an alternative source. There are 2 Smartcontract functions related.

The first function is the document registering function, the function act as part of the XDS Document Registry Actor to store the value of health document metadata interprets from ITI-42 transaction into the Blockchain ledger. This function act in a fashion similar to a programming variable where certain values were assigned to a specific variable for usage within the program, a whole set of metadata value is assigned into single Smartcontract function transaction. The function also automatically assigns each set of metadata with an identification number to be used for common understanding amongst the Smartcontract to differentiate each set of metadata belong to each health document. The identification number also essential for search operation which would be further explained later.

The second function is the checker function which will check for the last identification number assigned to the published set of metadata. This function allows Smartcontract to keep track of the identification number previously used and prevent duplication. Each time a new set of metadata entering the Blockchain ledger, the identification number which would be assigned to the metadata set will be additively increased by 1 from the previous.

**3.5.2 Document Search**

The search operation allows members of XDS Affinity Domain Blockchain to discover health documents existing in the network by searching for registered metadata set belong to the document within the Blockchain ledger and gain access to actual documents using access information provided in the metadata. There are 2 Smartcontract functions related.

The first function is the read function where Smartcontract allows the XDS Document Registry Actor to read the value of metadata stored within the Blockchain ledger. This function only needs identification number input to return metadata value to the XDS Document Registry Actor program.

During the search operation, the XDS Document Registry Actor will be the one to handle the search keyword. The Actor performs sequential searches on each set of registered metadata using the assigned identification number for iteration until the matching result was found or reached the end of the iteration. The Actor then triggers the return query result function.

The second Smartcontract function is the return query result function where the Smartcontract returns the whole set of metadata specified as the search result to the XDS Document Registry Actor. The version of the return value omitted by this function is different from the read function in terms of compatibility with the ITI-18 transaction format. The XDS Document Registry Actor then sorts the result into an ITI-18 transaction and return it to the XDS Document Consumer Actor.

Following the normal process of IHE XDS.b Profile in section 2.1.4, the XDS Document Consumer will then use health document access information provided within the search result metadata to gain access to the XDS Document Repository hosting the document. Then negotiate for document exchange using the ITI-43 transactions outside the Blockchain network.

As mentioned in section 3.2, after receiving the document shared from its original source, the shared peer will also want to register the document into the XDS Document Registry to let the network know that now they can act as an alternative source of the document for the network.

## **3.6 Process Flow**

*Figure 3‑3* showing the process flow of the XDS Document Registry Actor which acts as a medium between the Blockchain ledger and XDS System.

Diagram

Description automatically generated

Figure 3‑3 The process Flow of XDS Document Registry Actor

# 

**CHAPTER IV  
IMPLEMENTATION**

This chapter emphasizes the implementation step for this work. Begin with the setup method to establish a small Blockchain network for development which will describe the installation method and system specification. The second section then explains the implementation of each XDS Actor including its native programming code and corresponding Smartcontract detail. After that, the third section will explain the setup for the performance evaluation experiment and following with the experiment result in the last section.

For the implementation, the coding process was done within the main machine, which is a laptop Alienware 17 R5, Intel(R) Core (TM) i9-8950HK CPU@2.90GHz, 32 GB installed RAM, running on Windows 10 operating system version Home Single Language (20H2) 64-bit, x64-based processor. The code then deploys on the test machine which has simulated Blockchain environment setup within the virtual machine established within the main machine using Oracle VM VirtualBox as a virtual machine engine. The virtual machine running on Linux Ubuntu (64-bit) version 18.10 with 8 GB RAM and 100 GB storage dynamically shared from the host main machine.

## 4.1 Blockchain setup

For the implementation, the main machine was the best machine available for the implementation and compatible workflow. The Windows OS allows flexibility and provides access to a wide variety of software with a well-made GUI which is essential for coding and testing on unknown software. At the same time, as most Ethereum-related software is initially designed for Linux OS, so Ubuntu was selected OS for the test machine as it is a well-known Linux-based OS and available online at no cost. At the initial stage of development, there is Ubuntu version 16.05 available as the latest stable released. It was later upgraded to version 18.10. However, due to the community-based nature of the Ethereum platform, a software update can cause several issues interrupting the progression of the implementation. So, it was later fixed to a specified version to avoid conflict between each software component for the implementation.

Go-Ethereum or "Geth" client is the open sources software engine requiring to operate Ethereum Blockchain within each node. The client allows the user to issue commands to the node like initiate the Blockchain, start-stop mining/validating process for the Blockchain, and activate devp2p protocol to sync Blockchain data with other nodes. The client is available at [40]. Geth can be installed as standalone or included in the installation package of the Ethereum platform variant or other kinds of service interacting with Ethereum Blockchain (i.e., Ethereum Wallet Client). Geth's interface was initially designed to operate on Linux OS and later extend compatibility to other OS via Linux console simulated platform or work under graphic user interface of another client. The Geth client has no specific system requirement as it only is a set of Golang scripts that has no restriction to any system but, as it is mostly integrated within another client, so it requires the machine to compatible with the main client for installation. In the implementation, the test machine has Geth globally installed by installing Quorum, so the client can be launched from anywhere regardless of environment path.

As mentioned in section 2.3.4.2, Quorum is an Ethereum-forked that allows the Blockchain to adopt a consensus mechanism other than PoW and PoS which default to Ethereum Blockchain. Quorum's source code and installation package can be accessed at [31]. The platform was designed to specifically operate with a Linux-based interface and can be compatible with non-Linux OS with the aid of 3rd party software as a medium. Other than that, the platform has no specific system requirement for installation. However, from the test during the implementation, it is recommended that the machine running Quorum should have more than 6 GB of available RAM. Otherwise, there will be a performance issue that occurred during the run. In this implementation, as the test machine running on Ubuntu 18.10, there is no other 3rd party software required to operate Quorum. The Geth client was included in the Quorum installation package, which means a developer can immediately start their Quorum Blockchain development right after the installation.

Other than the Blockchain platform, Quorum also offers the "7-Nodes Example" for developers to locally deploy in their machine to test the functionality and performance of Smartcontract during the development. The source code can be cloned directly from its repository available in Github [33]. The "7-Nodes" will simulate seven Blockchain nodes in the host machine in a similar fashion to a virtual machine using the required library called "Tessera" [41] and "Constellation" [42] included in the package as a running engine. The source code can be initiate, activate, and simultaneously controlled using the control script provided within the example. Each node can be accessed using the Geth client. For the implementation, the control script provided within the example was further modified to be compatible with our usage. The Blockchain initiation script was modified to be able to re-initiate the entire Blockchain by deleting the existing chain and replace with the empty one. This allows reset of published Blockchain during the development. The transaction publishing script was modified to run other specific code developed for XDS Document Registry Actor and its Smartcontract, allow testing and running of XDS Document registry-related code on the 7-Nodes. It must be noted that transaction-related scripts only operate on specified single nodes amongst the seven, not the entire set of the seven nodes. That means the activity of each node is independent.

- Configure initial Block for the Blockchain

- Script running 7-Nodes

For Smartcontract programming, Ethereum providing a web-based IDE for Solidity language that can compile, testing, and deploy smart-contract to specific Ethereum node called “Remix” [43]. The Remix IDE has no specific system requirements nor versions of its own, but it requires a Web Browser to access as it is a web-based IDE. the Web Browser running the IDE must be up-to-date to provide an essential Website environment. At the same time, the machine running the Web Browser must provide enough memory space for the Web Browser to maintain the IDE without an issue. Within the Remix IDE, they also provide a wide variety of community-based plugins for Solidity developers to aid in code compiling, debugging, and test deployment on a simulated Blockchain environment. The IDE also offers a choice to connect directly to an active Ethereum Blockchain network whether it is the main Ethereum Blockchain or local chain under development. Additionally, each session of the Remix coding workspace is saved using a unique URL and can be saved locally using Remix local client installed in the machine. In the implementation, Remix was accessed using Google Chrome from the main machine and the session was saved locally using Remix client to avoid unexpected issues on the Solidity code. Each Smartcontract is validated, and test deployed within the IDE before actual implementation on the test machine. The Smartcontract ready for implementation will be compiled using the Solidity compiler provided by Remix community-based plugin. Each compiled solidity code gives ABI code and Binary code which will be used on Web3js code to interact with the Blockchain Smartcontract.

## 4.2 XDS Actors

As we have seen from HL7 and FHIR, current healthcare information exchanged related standards are majorly web-based protocol. Additionally, development of IT infrastructure to support healthcare operation require the capability to handle a huge amount of transaction in a limited amount of time so, it requires our system implementation to be able to handle multitask properly. With asynchronous nature and compatibility with website integration, Javascript is one of the best choices for our implementation of this work. In this implementation, we adopt the "Node.js" variant of Javascript as it was made to build scalable network applications that handle many connections concurrently. Furthermore, Node.js also providing simple access to community-made node modules which offer a wide variety of useful APIs for software development which may reduce difficulty in our implementation further.

All actors within IHE XDS Profile communicate with each other using XML message transaction. As we utilize Javascript as main programming language for the implementation, these XML messages need to be interpreted into programming object to allow simpler handling method within the program. Javascript Object Notation (JSON) is a lightweight data-interchange format of programming object which was invented to serve the purpose. It is easy for humans to read and write and easy for machines to parse or generate. That mean, all XML message transactions sent to XDS Document Registry actor program will be converted into JSON. For this implementation, we utilize NodeJS “xml2js” module for the task.

To interface our program to Ethereum smart contract, we can use Ethereum API tools which is Web3 [44] as a middle. Web3 allows smart contract control through preferred programming language and transitions logic and variables from the language to Solidity. Web3 provided a programming API for Javascript called "Web3JS" which allows the Javascript program to interact with Ethereum based smart-contract. The API can be accessed using the node module provided via Node.js.

### 4.2.1 XDS Document Repository Actor

4.2.1.1 Interpret IHE ITI-42 transaction

Figure IV A and Figure IV C, showing XML language code snippet of Registry Document Set-b [ITI-42] transaction sample. The code composing of 2 main sections. The first section labeled with “lcm:SubmitObjectRequest” is where XML schematic information are located and the label also act as marker which tell interpreter program to recognize it as ITI-42 transaction. The second section start from label “rim:RegistryObjectList” following with “rim:ExtrinsicObject” contain all information regarding corresponding health document. This section is where all Metadata attributes of the document are located. If the Document Registry Actor successfully received the transaction, they must return response as shown in Figure IV B. The response transaction included only XML schematic information, message UUID number, and status type “successful”. This response will let the Repository finish its process and end messaging attempt.



Figure 4‑1 Pseudocode represents general format of Register Document Set-b [ITI - 42]



Figure 4‑2 XML Code snippet of Registry Document Set-b Response transaction sample













Figure 4‑3 XML Code snippet of Registry Document Set-b [ITI-42] transaction sample

4.2.1.2 XDS Document Repository Actor simulating program

For document registering, XDS Document Repository Actor register document Metadata attributes into XDS Document Registry Actor using IHE ITI-42 transaction. XDS Document Registry Actor then interprets the transaction into a programmable object before check if the transaction is ITI-42. Then, the actor proceeds to pass the retrieved object into Blockchain smart-contract and publish it into a Blockchain ledger.



Figure 4‑4 Javascript Code Snippet of XDS Document Repository Actor

### 4.2.2 XDS Document Consumer Actor

4.2.2.1 Interpret IHE ITI-18 transaction

Figure IV G, showing XML language code snippet of RegistryStoredQueryRequest [ITI-18] transaction sample. The code composing of   
3 main sections. The first section labeled “query:AdhocQueryRequest” is where XML schematic information are located and the label also act as marker which tell interpreter program to recognize it as ITI-18 transaction. The second section labeled “query:ResponseOption” mark the expected format of query result that will return to Document Consumer. The third section start from label “rim:AdhocQuery” contain all search keywords issued by Document Consumer. These search keywords are selected Metadata attributes and its value. When Document Registry Actor received the transaction, they will use search keyword provided to search for registry with matched Metadata attributes value then return the result to Document Consumer Actor as response transaction following Figure IV H. With header labeled “query:AdhocQueryResponse”, the transaction contain search result depend on query type specified in ITI-18 transaction. If the query expected for “LeafClass” as result, the response would return Metadata attributes of all matched result in detailed. Otherwise, if the query expected for “ObjectList”, the response would return object reference number of all matched result. These two types of response specifically selected depend on search behavior of Document Consumer Actor’s user. The query which specified “LeafClass” as its search result must provide keyword which unique to its corresponding document, such as document unique ID or object reference UUID. At the same time, “ObjectList” are used to search for wide range of document with generic search keyword and value where discovery of document existent is the main goal.



Figure 4‑5 Pseudocode represents general format of Registry Stored Query Request [ITI - 18]



Figure 4‑6 Pseudocode represents general format of Query Response   
included “Object Reference” of search results





Figure 4‑7 Pseudocode represents general format of Query Response   
included "Leaf Class" of search result



Figure 4‑8 XML Code Snippet of RegistryStoredQueryRequest [ITI-18] Transaction Sample











Figure 4‑9 XML Code Snippet of RegistryStoredQueryResponse Transaction Sample

4.2.2.2 XDS Document Consumer Actor simulating program

For document query, XDS Document Consumer query for document Metadata attributes stored within XDS Document Registry Blockchain providing search operation type and some Metadata attributes value as search keyword via ITI-18 transaction. XDS Document Registry will check if the transaction is ITI-18 before performing search operation matching specified search type using provided keyword Metadata attributes value. The search operation will be performed by consequently call for each registered smart-contracts until all contracts with matched attributes value were found. XDS Document Registry Actor then returns all query result in XML format following specification for ITI-18 responding. Upon receiving the query response, XDS Document Consumer then interprets the transaction and displays the result to the user in a human-understandable format.

Following IHE XDS Profile, XDS Document Consumer actor is where the user specifies search keyword values of Metadata attributes for system to query for matching document exist within XDS Affinity Domain. For this implementation, we design that user interface will take a form of command line program that can be run via Windows command prompt or Linux terminal. The program will prompt the user to specify search type, including META-attributes value, and specify the value. The actor then accepts these values to create XML message following ITI-18 format before sending it to local or accessible XDS Document Registry actor to query for matching document and start search operation.



Figure 4‑J Javascript Code Snippet of XDS Document Consumer Actor

### 4.2.3 XDS Document Registry Actor

The XDS Document Registry Actor program must be able to communicate with simulated XDS Document Repository actor and XDS Document Consumer actor. At the same time, the software will need to act as the middle between local XDS system and the Blockchain ledger. When ITI-42 was interpreted into JSON, the actor then passes the object into smart-contract. For the implementation, smart-contract was designed to store string value and will return the stored value when called by Geth client. The prepared JSON must be converted into string before entering smart-contract. This is due to limit of Ethereum smart-contract which can cover limit number of programming variable so, we simplify our program to avoid that limit by storing whole JSON in string form as single variable. However, because of Ethereum Blockchain require certain amount of gas to execute smart-contract, the length of the variable may cause error in the process if there was not enough gas supplied. That mean, we need to increase limit amount of gas for executing smart-contract from default value. Although, this change is not affecting this implementation for concept demonstration but, it may affect the network where its member prefers to use actual cryptocurrency like Ether to maintain Blockchain. This may accelerate depletion of currency circulating in the network and severe maintainability of the chain. By these smart-contract design, XDS Document Registry actor can keep Metadata attributes of each document by store it as JSON string variable inside Blockchain using one smart-contract per document. At the same time, the actor can perform search operation by sequentially call upon each published smart-contract one-by-one until the result was found or until the last in the case which no matching result. Publishing of smart-contract require gas to execute while calling smart-contract not consuming Blockchain resource.

add pic or pseudocode

Introduce to 4.2.3.1 to avoid confuse (separated by function instead)

4.2.3.1 Implementing Document Register function

4.2.3.1.1 Native code

IHE ITI-42 is XML format transaction used for registering Metadata attributes of new document storing in XDS Document Repository actor into XDS Document Registry actor. The transaction specified with header “RegisterDocumentSet-b” and compose of Metadata attributes of corresponding document. The content of the attributes varies upon type of document and the event represent by the document. For this implementation, XDS Document Registry actor will open TCP connection to receive the transaction on specified port. After ITI-42 transaction is received, the actor then converts XML message into JSON using xml2js. After that, the program extracts Metadata attributes and prepare it for smart-contract.

Figure 4‑4 showing mockup code snippet of XDS Document Repository actor who simply read mockup ITI-42 transaction then send the message to XDS Document Registry. The Repository will wait for response from the Registry before finishing its process. For the demonstration of this work, we created several mockup ITI-42 transactions by changing Metadata attribute values.



Figure 4‑K Javascript Code Snippet of XDS Document Registry Actor  
Node Module import declaration and TCP Socket message receiver section



Figure 4‑L XDS Document Registry Actor  
This section checks if receiving message is ITI-42 or ITI-18 identified by its header



Figure 4‑M XDS Document Registry Actor  
Declaration of JSON variable to store all Metadata attributes by its position in the format



Declaration of JSON variable to store all Metadata attributes by its position in the format (Continued)



Figure 4‑N XDS Document Registry Actor  
Define variable of each Metadata attribute UUID label following IHE ITI Framework

















Figure 4‑O XDS Document Registry Actor  
This section interpret and assort Metadata attribute value from ITI-42 to JSON







Figure 4‑P XDS Document Registry Actor  
This section passes JSON into Smart Contract as single string variable

4.2.3.1.2 Smartcontract

Smart-contract was developed to store programming logic or algorithm as blockchain transaction. These smart-contract transactions can be compiled by Ethereum client which will give the result of its script or code (for example, read or return specific value). So, we design smart contract which when executed, it will spawn smart contract that stores given document Metadata attributes value within number labeled smart-contract instances which encoded in Blockchain transaction. When these instances were called, it will return the stored metadata attributes value back. Allow the search program to identify the set. At the same time, this allows document registry to store within Ethereum Blockchain. These composed to function as Document Registry Smart Contract. Due to limit of Smart Contract that it cannot hold variable more than 15 variables and its size are limited by supplied gas so, in Figure IV I, we only utilize Smart Contract to act as simple text storage without the usage of many variables. Convert the entire set of Metadata attributes into a single string variable, eliminates the need for too many variables inside Smart Contract. The Contract then simply receives the string variable when the ‘store’ function was invoked and returns the stored string variable value when the ‘retrieve’ function was invoked. There is 2 form of Metadata attributes to be stored in Smart Contract. The first form is assorted Metadata attributes JSON achieved from the XML interpreter program. This form will be used by the search program for matching with search keywords input from Document Consumer. The second form is raw XML Metadata attributes which will be returned as query result to Document Consumer. This maintains the compatibility with the traditional XDS system which Document Registry would simply return Metadata attributes from its database in XML format. Additionally, to avoid data misplacement in Smart Contract, each contract will be issued with a numeral document id to keep track number of documents stored in Blockchain Smart Contract. The search program can then check for a number of the latest document Metadata attributes published into Blockchain Smart Contract and perform a sequential search started from the first to the last documents.



Figure 4‑17 Solidity Code Snippet of Smart Contract used in this work

4.2.3.2 Implementing Document Search function Smartcontract

4.2.3.2.1 Native code

Similar with IHE ITI-42 transaction handling, XDS Document Registry actor also wait for ITI-18 on TCP channel. The received transaction will be converted into JSON. The transaction specified with header “QueryResponse” and compose of Metadata attributes value input by Document Consumer. These values will be used in search operation which will seek for the smart-contract with matching Metadata attribute values. After the result was found, the actor then proceeds to create response XML message following the format provided by IHE ITI Framework.



Figure 4‑R XDS Document Registry Actor  
Define variable of query request type UUID label following IHE ITI Framework



Figure 4‑S XDS Document Registry Actor  
Identify query request type following received ITI-18 header and assort search keyword



Identify query request type following received ITI-18 header and assort search keyword (Continued)



Identify query request type following received ITI-18 header and assort search keyword (Continued)



Identify query request type following received ITI-18 header and assort search keyword (Continued)



Figure 4‑T XDS Document Registry Actor  
Check for the latest document ID published in Blockchain before beginning search operation



Check for the latest document ID published in Blockchain before beginning search operation  
(Continued)



Check for the latest document ID published in Blockchain before beginning search operation  
(Continued)



Figure 4‑U XDS Document Registry Actor  
Begin search operation by sequentially check each published contract one-by-one



Begin search operation by sequentially check each published contract one-by-one (Continued)



Begin search operation by sequentially check each published contract one-by-one (Continued)



Figure 4‑V XDS Document Registry Actor  
Check if value of Metadata attributes in each publish contract matched with search keyword before summarize search result.



Check if value of Metadata attributes in each publish contract matched with search keyword before summarize search result (Continued)



Check if value of Metadata attributes in each publish contract matched with search keyword before summarize search result (Continued)



Check if value of Metadata attributes in each publish contract matched with search keyword before summarize search result (Continued)



Check if value of Metadata attributes in each publish contract matched with search keyword before summarize search result (Continued)



Figure 4‑W XDS Document Registry  
Gather search result and response back to Document Consumer Actor



Gather search result and response back to Document Consumer Actor (Continued)

4.2.3.2.2 Smartcontract

As required in IHE ITI Framework, Document Registry Actor must be able to respond to the query from the Document Consumer by returning the Metadata attributes of the registered document matched with the query to the consumer. In a traditional database, this can be done by utilizing a query of a relational (SQL) database. However, for Blockchain, the structure of stored data is different from relational database but similar to NoSQL. That mean, search operation will need to rely on a sequential search algorithm. The program will need to take a look at all published transactions one-by-one from the first until the result was found. Each transaction will require the program to call on smart-contract for reviewing the stored value before comparing it with the specified value used for search. When all of the values called from the smart-contract are matched with the value specified for search, the value called will be marked as a search result which will be returned to XDS Document Consumer Actor via ITI-18 format.

## 4.3 Experimental setup

As to test if this implementation can operate with XDS Actors in a common XDS system, we use transaction samples provided by the IHE ITI framework with modified attributes value to evaluate the system. Transaction samples provided by the framework are including ITI-42 Register Document Set-b transaction, ITI-18 Registry Stored Query transaction and its corresponding response transaction. However, these transaction samples have limited capabilities as an example due to much more specification provided in the framework so, there is some transaction need to be defined manually from requirements provided in the framework.

In this implementation, we evaluate the result concept demonstration by measuring performance of each major process. This will reflect compatibility of the concept to healthcare operation environment which require continuous and huge amount of process at short amount of time.

## 4.4 Result

### 4.4.1 Result of each step

- This one should be the result i.e., screen capture

- XML to JSON contribution

**4.4.2 Performance Evaluation Result**

The evaluation measure time the program needs to finish each process, consist of registering document into XDS Document Registry Actor Blockchain and when XDS Document Consumer Actor query for the registered document using specified keywords. It took average 4.797 ms for ITI-42 to be sent from XDS Document Repository Actor to XDS Document Registry Actor locally using TCP connection. XDS Document Registry Actor took an average of 5 seconds and 174.691 ms to the published transaction into Blockchain. With a minimum number of keywords, it took an average of 221.884 ms to finish search operation on Blockchain with 10 samples smart-contract published beforehand while it took an average of 260.480 ms for XDS Document Consumer Actor to received query response after the query was sent to XDS Document Registry Actor. With a maximum number of keywords, it took an average of 264.937 ms to finish search operation on Blockchain with 10 samples smart-contract published beforehand while it took an average of 304.457 ms for XDS Document Consumer Actor to received query response after the query was sent to XDS Document Registry Actor.

**CHAPTER V  
CONCLUSION AND DISCUSSION**

This chapter concluding the work.

**5.1 Answering Problem Statement**

XDS Registry Blockchain allow sharing of healthcare document between different healthcare organizations which require maintain of its confidentiality while mitigate emerging cyber-threats on healthcare domain that tamper with integrity and availability of data, there need document registry that have distributed, decentralized, persistent, and immutable characteristics.

**5.2 Implementation Source Code**

We are providing source code of the implementation at (https://github.com/semiangel/XDSchain.git). However, must be noted that this implementation relies on virtual

nodes provided by Quorum “7 Nodes Example”. Any implementation on additional physical node or separated virtual machine may require additional adjustment on network connection between nodes.

**5.3 Limit of implementation based on current technology**

**5.4 Suggestion for concept adoption**

**5.5 Suggestion for future work**

**REFERENCE**

[1] Weinelt B. Digital Transformation of Industries. Logistics Industry, http://reports.weforum.org/digital-transformation/wp-content/blogs.dir/94/mp/files/pages/files/digital-enterprise-narrative-final-january-2016.pdf (2016).

[2] Marcelo A, Medeiros D, Ramesh K, et al. Transforming Health Systems Through Good Digital Health Governance. *adb Sustain Dev Work Pap Ser* 2018; 1–15.

[3] Shaw T, Hines M, Kielly C. *Impact of Digital Health on the Safety and Quality of Health Care*, https://www.safetyandquality.gov.au/wp-content/uploads/2018/02/Report-The-Impact-of-Digital-Health-on-Safety-and-Quality-of-Healthcar....pdf (2000).

[4] Cisco. The Digitization of the Healthcare Industry: Using Technology to Transform Care. *Cisco* 2017; 1: 12.

[5] Bullhound G. *Digital healthcare*. 2015.

[6] Meskó B, Drobni Z, Bényei É, et al. Digital health is a cultural transformation of traditional healthcare. *mHealth* 2017; 3: 38–38.

[7] Carestream Health. Interoperability : Connecting the Healthcare Enterprise to Deliver Responsive Patient Care. 2015; 1–9.

[8] PolicyMedical. Interoperability in Healthcare: To Have or Not to Have, https://www.policymedical.com/interoperability-healthcare/ (accessed 22 September 2018).

[9] Interoperability DH. Digital Healthcare Interoperability.

[10] Healthcare Information and Management Systems Society. Definition of Interoperability. *Himss* 2013; 2013.

[11] Oracle. Interoperability : A Key to Meaningful Use. *Solutions*, http://www.oracle.com/us/industries/healthcare/interoperability-wp-188782.pdf (2010).

[12] HIMSS. What is Interoperability?, https://www.himss.org/library/interoperability-standards/what-is-interoperability (accessed 27 April 2019).

[13] Paige Goodhew. Why Healthcare Interoperability Matters | Redox, https://www.redoxengine.com/blog/why-healthcare-interoperability-matters/ (accessed 27 April 2019).

[14] Dr.David Hay. Why is interoperability so important for healthcare organisations? | Orion Health, https://orionhealth.com/global/knowledge-hub/blogs/why-is-interoperability-so-important-for-healthcare-organisations/ (accessed 27 April 2019).

[15] Le Bris A, Asri W El. STATE OF CYBERSECURITY &amp; CYBER THREATS IN HEALTHCARE ORGANIZATIONS Applied Cybersecurity Strategy for Managers. *ESSEC Bus Sch* 2017; 13.

[16] Healthcare IT News. The biggest healthcare breaches of 2017, https://www.healthcareitnews.com/slideshow/biggest-healthcare-breaches-2017-so-far?page=1 (accessed 11 September 2018).

[17] HIPAA Journal. Largest Healthcare Data Breaches of 2018, https://www.hipaajournal.com/largest-healthcare-data-breaches-of-2018/ (accessed 27 April 2019).

[18] Healthcare IT News. The biggest healthcare data breaches of 2018 (so far), https://www.healthcareitnews.com/projects/biggest-healthcare-data-breaches-2018-so-far (accessed 27 April 2019).

[19] IHE International Inc. About IHE, https://www.ihe.net/about\_ihe/ (accessed 11 September 2018).

[20] IHE International Inc. IHE Process, https://www.ihe.net/about\_ihe/ihe\_process/ (accessed 11 September 2018).

[21] IHE International Inc. Profiles, https://www.ihe.net/resources/profiles/ (accessed 17 September 2018).

[22] IHE International Inc. IHE IT Infrastructure ( ITI ) Technical Framework Volume 1 Integration Profiles. *Int J Healthc Technol Manag* 2008; 1: 1–177.

[23] dkorolyk. What Is The Difference Between XDS,XDS.a,XDS.b and XDS-I?, http://healthcareitsystems.com/2012/05/22/what-is-the-difference-between-xds-xds-a-xds-b-and-xds-i/ (2012, accessed 17 February 2019).

[24] Luke MN, Lee SJ, Pekarek Z, et al. Blockchain in Electricity: a Critical Review of Progress to Date. 2018; 1–36.

[25] PwC. a Catalyst for New Approaches in Insurance.

[26] Zheng Z, Xie S, Dai H, et al. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. *Proc - 2017 IEEE 6th Int Congr Big Data, BigData Congr 2017* 2017; 557–564.

[27] Yaga D, Mell P, Roby N, et al. Blockchain Technology Overview (NISTIR-8202). *Draft NISTIR* 2018; 59.

[28] Buterin V. *A NEXT GENERATION SMART CONTRACT & DECENTRALIZED APPLICATION PLATFORM*.

[29] ethereum/devp2p: Ethereum peer-to-peer networking specifications, https://github.com/ethereum/devp2p (accessed 3 June 2021).

[30] Solidity Programming Language | The Solidity language portal is a comprehensive information page for the Solidity programming language. It features documentation, binaries, blog, resources & more., https://soliditylang.org/ (accessed 3 June 2021).

[31] Quorum. GoQuorum, https://github.com/ConsenSys/quorum (2020, accessed 3 June 2021).

[32] Morgan JP. Quorum | J.P. Morgan, https://www.jpmorgan.com/global/Quorum (2017, accessed 26 April 2019).

[33] Quorum 7-nodes Example, https://github.com/ConsenSys/quorum-examples/tree/master/examples/7nodes (accessed 3 June 2021).

[34] Peterson K, Deeduvanu R, Kanjamala P, et al. A Blockchain-Based Approach to Health Information Exchange Networks. *Mayo Clin* 2016; 10.

[35] Ekblaw A, Azaria A, Halamka JD, et al. A Case Study for Blockchain in Healthcare: " MedRec " prototype for electronic health records and medical research data. *IEEE Technol Soc Mag* 2016; 1–13.

[36] Zyskind G, Nathan O, Pentland AS. Decentralizing privacy: Using Blockchain to Protect Personal Data. *Proc - 2015 IEEE Secur Priv Work SPW 2015* 2015; 180–184.

[37] Li H, Zhu L, Shen M, et al. Blockchain-Based Data Preservation System for Medical Data. *J Med Syst* 2018; 42: 1–13.

[38] Tanwar S, Parekh K, Evans R. Blockchain-based electronic healthcare record system for healthcare 4.0 applications. *J Inf Secur Appl* 2020; 50: 102407.

[39] Sultan K, Ruhi U, Lakhani R. *CONCEPTUALIZING BLOCKCHAINS: CHARACTERISTICS & APPLICATIONS*, https://arxiv.org/ftp/arxiv/papers/1806/1806.03693.pdf (2018, accessed 29 October 2018).

[40] ethereum/go-ethereum: Official Go implementation of the Ethereum protocol, https://github.com/ethereum/go-ethereum (accessed 5 June 2021).

[41] ConsenSys/tessera: Tessera - Enterprise Implementation of Quorum’s transaction manager, https://github.com/ConsenSys/tessera (accessed 5 June 2021).

[42] ConsenSys/constellation: Peer-to-peer encrypted message exchange, https://github.com/ConsenSys/constellation (accessed 5 June 2021).

[43] Remix - Ethereum IDE, https://remix.ethereum.org/ (accessed 8 March 2021).

[44] web3.eth.Contract — web3.js 1.0.0 documentation, https://web3js.readthedocs.io/en/v1.3.4/web3-eth-contract.html (accessed 8 March 2021).

**APPENDICES**

**APPENDIX a**

**ETHICAL APPROVAL DOCUMENT**

**APPENDIX**

**ETHICAL APPROVAL DOCUMENT**

**BIOGRAPHY**

|  |  |
| --- | --- |
| **NAME** | Petnathean Julled |
| **DATE OF BIRTH** | 22 October 1992 |
| **PLACE OF BIRTH** | Nakhon Si Thammarat, Thailand |
| **INSTITUTIONS ATTENDED** | Mahidol University, 2011-2021 |
|  | Bachelor of Engineering (Biomedical) |
|  | Mahidol University, 2011-2015 |
|  | Master of Science (Cybersecurity and Information Assurance) |