2017

A Model for improving interoperability of healthcare systems in a distributed environment

Benson Wandera Ogutu
Fuculty of Information Technology (FIT)
Strathmore University

A Model for Improving Interoperability of Healthcare Systems in a Distributed Environment

Benson Wandera Ogutu

046369

Masters of Science in Information Technology

2017

A Model for Improving Interoperability of Healthcare Systems in a Distributed Environment

BENSON WANDERA OGUTU

Submitted in partial fulfillment of the requirements for the Degree of Masters of Science in Information Technology at Strathmore University

Faculty of Information Technology

Strathmore University

Nairobi, Kenya

June, 2017

Declaration

I declare that this work has not been previously submitted and approved by Strathmore University or any other university. To the best of my knowledge and belief, this dissertation contains no material previously published or written by another person except where due reference is made.

Benson Wandera Ogutu

Approval

The Thesis of Benson Wandera Ogutu was reviewed and approved by the following:

Dr. Vincent Oteke Omwenga,

Senior Lecturer, Faculty of Information Technology,

Strathmore University.

Dr. Joseph Orero

Dean, Faculty of Information Technology

Strathmore University.

Prof. Ruth Kiraka,

Dean, School of Graduate Studies,

Strathmore University.

Abstract

Access to patient health record for a patient from one healthcare institution to another has had its fair share of challenges. The two healthcare institutions under study have got their own distributed healthcare systems but none of these institutions can access or share their patient health records across. This has hugely been down to the complexity of the healthcare domain, standardization challenges, legacy systems, legal challenges, resistance to change, privacy and security. The study developed an interoperability model for improving patient health records access and sharing across distributed healthcare systems. The modelled application allows two or more distributed healthcare systems to access and share patients' health records. This model tries to work around the challenges identified above making the system an open system such that any healthcare system can be plugged into it and facilitate data sharing and access. The study applied agile software methodology as it allows for faster iterations and frequent release while factoring in user feedback. The modelling of interoperable distributed healthcare systems is of great importance as it allows for ease of access, portability of data, data confidentiality, integrity and security, capture of data in different formats, file sharing, reduction of costs both to the patients and healthcare institutions and makes the systems robust and scalable.

Keywords: Healthcare Institutions, API, Interoperability

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to specifically acknowledge my wife for all the support given and proof reading my document. I applaud my colleague, Richard Omoka for the advice throughout the project especially regarding the API design. I thank Dr. Vincent Omwenga for the noble sacrifice you have for me and the vast knowledge I have learnt both relating to this project and outside it.

Dedication

I dedicate this dissertation to my beloved God, my wife, family, Congress WBN, classmates and friends for the support they have given me throughout this dissertation period. I am forever grateful to Dr. Vincent Omwenga for patiently walking with me through this endeavor and the entire Strathmore University family.

Chapter 1 : Introduction

1.1 Background to the study

Interoperability in distributed healthcare systems (DHS) has not been realized mainly because of the existence of autonomous distributed healthcare systems used within the healthcare organizations and developed using various programming languages, tools, data formats, syntax and semantics among others to record and present patient information. Failure to realize interoperability has hampered the seamless exchange of meaningful patient health records across various distributed healthcare systems owing to the existence of health records in various data formats. Interoperability in DHS can allow healthcare practitioners (e.g. Doctors) to have real-time access to patient information thereby allowing them to offer timely & quality services, make informed decisions and reduce costs for the patients (The Regenstrief Institute, 2012). This study therefore looks at interoperability in DHS, analyzes interoperability challenges in distributed healthcare systems and further develops an API-Based model for Improving Interoperability of Healthcare Systems in a Distributed Environment.

1.2 Distributed Healthcare Systems

Distributed systems are said to be a collection of independent computers or systems that appears to its users as a single coherent system (Tanenbaum & Maarten, 2007). A distributed healthcare system can be said to comprise of autonomous finance, pharmacy, imaging, patient administration and hospital management modules or functional areas. Each of this functional areas can have a system developed to aid in achieving their functional goals. To mitigate data replication in the event that a patient has to visit all these functional areas, a platform that allows for unified access with access controls is enforced to make the modularized systems to appear as a single coherent system. The main objectives of a distributed system include accessibility of resources, transparency, openness and scalability (Tanenbaum & Maarten, 2007).

Distributed system allows users and other systems to access resources from anywhere while allowing them to share the information in an organized and efficient manner. In comparison with a centralized system, a distributed system ensures system availability in the event that a system downtime occurs. Centralized systems provide single point of failure unlike distributed system (Tanenbaum & Maarten, 2007). Resources in this study can be patient's health records, drug supply and drug use information among others. This has economic benefit in that it is cheaper for a patient to have his or her health records accessed from anywhere such that a physician can attend to the patient from any healthcare institution. Accessibility also allows for increased collaboration and information exchange that is a doctor in Healthcare Institution A can freely share information with a doctor in Healthcare Institution B. This makes the doctors highly effective while reducing the delays and traveling a patient has to make to seek for treatment from one Healthcare Institution to another. Access to available resources carries with it the security challenges. Healthcare organizations handle patient information which is sensitive and should be treated as such. There is a greater need to enforce security so that patient information is only accessed by those authorized to access it else it will be compromised leading to negative impact both on the patient and the Healthcare Institution.

Transparency of a system lies in its ability to hide the actual system distribution process from its end users. Users of the systems and other applications must interact with a distributed system as though it were a single system. A patient visiting Healthcare Institution A and then is transferred to Healthcare Institution B should feel as though it is one and the same Healthcare Institution since what is available in Healthcare Institution A is what is in Healthcare Institution B. Below is a summary on the different types of transparencies that exist:

Table 1.1: Types of Transparencies in a distributed system.

Transparency	**Description**	**Outcome**
Access	Hide data differences in data representation and how a resource is accessed.	Define how data is to be represented.
Locations	Hide where resource is located	Define the naming structure. Logical names should hide the actual resource.

Migration	Hide that a resource may move to another location	Allow for movement of resources without affecting how they are accessed.
Relocation	Hide that a resource may be moved to another location while in use.	There must be no disconnect.
Replication	Hide that a resource is replicated.	Increased availability and improved performance.
Concurrency	Hide that a resource may be shared by several competitive users	Resource must remain in a consistent state.
Failure	Hide the failure and recovery of a resource.	Ability to recover from failure.

Openness demonstrates the ability of a system to allow services to be offered based on defined rules or guidelines that describe the syntax and semantics belonging to the services in distributed systems. Established widely accepted standards used for syntactic standardization in healthcare include HL7/CDA; DICOM; and EDIFACT (Pedersen & Hasselbring, 2004). Semantic standards derived via classification or terminology system are required for the encoding of healthcare data (Sunyaev, Leimeister, Schweiger and Krcmar, 2008). These standards are ICD, SNOMED, MeSH, UMLS, and LOINC.

A system is said to be scalable based on its size i.e. its ability to cater for additional users and resources, geography e.g. distance between the system, users & resources and administration e.g. system administration in the different administrative organizations (Neumann, 1994). A system exhibiting more than one of these dimensions loses certain levels of performance as it scales up.

1.2.1 Interoperability in Distributed Healthcare Systems

The study adopts HIMSS definition of interoperability as being the ability of different information technology systems and software applications to communicate, exchange data, and effectively allow patients and healthcare practitioners to use that information (HIMSS, 2013). Interoperability has been a challenge owing to the existence of heterogeneous distributed systems that ought to communicate with one another, exchange of data that exists in different data formats in the different distributed systems and finally the need to have patients and healthcare practitioners access the patient information from anywhere. The study notes that the underlying mechanisms for data sharing and exchange must be hidden to the user of the system such that one does not need to know the system from which they are accessing the information. The availability of the healthcare system is vital for the availability of the data. This data must be accessible if and when required.

Figure 1.1 describes interoperable processes flow in a distributed healthcare system:

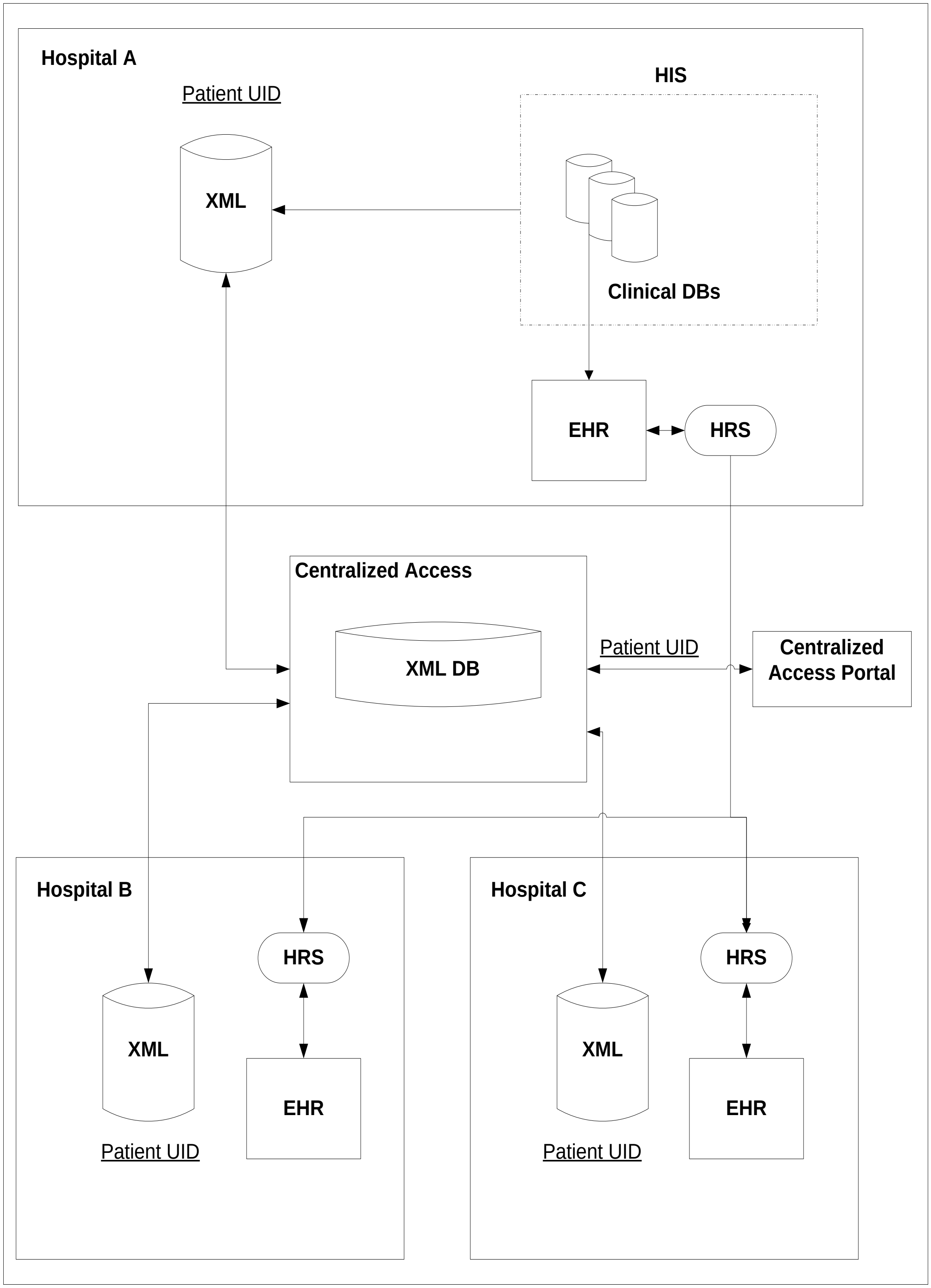

Figure 1.1: Interoperable Distributed Healthcare (Adapted from Tanenbaum & Maarten, 2007)

The clinical databases contained in the HIS module stores patient information and it allows healthcare practitioner to interact with the Interoperable Distributed Healthcare System. For online operations, healthcare practitioner and patients are required to enter a patient UID, which universally identifies the patient. This allows healthcare practitioner to access, edit and update patient's initial and current diagnoses whereas for the patient, one can only view the information available in the system. HRS is a service that scans to see whether a patient's record exists in a system or not based on the UID. In the event that a patient record exists, the patient or practitioner will proceed to access it else it will indicate no record found. For centralized access into the interoperable portal, a centralized access DB is developed and allows one to access information by keying in a UID. The UID should thus provide patient's information like, patient's details, last time of visit to a healthcare facility, name of healthcare facility visited, diagnosis and medication. The patient UID allows the DB to query only visited hospitals as opposed to all the hospitals available (Shaker & Samir, 2013).

1.2.2 Challenges facing interoperability in distributed healthcare systems

Interoperability implementation in distributed healthcare systems has undergone major challenges which include Complex Healthcare Domain, Standardization challenges, Legacy Systems, Legal challenges, Resistance to Change and Privacy and Security (Iroju, Soriyan, Gambo and Olaleke, 2013).

The complexity of the healthcare domain arises from the various actors and processes a patient has to go through during the treatment process. Each of the actor has inputs into the system that acts as information for other actors e.g. a lab technician's laboratory examination report is to be viewed by the doctor to determine the cause of a problem and eventually mode of treatment to be administered. The actors in the system include, doctors, nurses, lab technicians, radiologists, physiotherapists, and pharmacists. Iroju et al. (2013) identifies the different types of data to include administration of patients, organization information, laboratory and clinical data. The ability to exchange this information across platforms and systems while maintaining data confidentiality, integrity and availability ensures its safety and enhances effectiveness in healthcare. Presence of in-house or off-the-shelf software comes with the compatibility challenge when information sharing is required making such systems not be interoperable with others.

While discussing openness as one of the objectives of a distributed systems, the study highlighted the different standards available categorized as either semantic or syntactic standards. It must be noted that standards are subject to different interpretations which means that system development will vary from one organization to the next making it hard to have them achieve interoperable. Over the years, systems have undergone massive transformation with updates on syntactic and semantic standards while systems that existed before the updates still in use. No major updates or upgrades have been made on the systems hence the term "Legacy". Implementation of such systems and other customized systems make interoperability a nightmare especially in the event that the systems were developed without any determined standards as discussed above (Iroju et al, 2013).

As the study discusses interoperability challenges, legal challenges must be discussed. Exchange of information must maintain the CIA of data or information else this could lead to legal tussles owing to the breach of confidentiality of patients' information. As a patient's data is keyed in the system, one has confidence that the information will only be available actors and specifically for treatment purposes. Any action outside this necessitates legal action. As with many other systems, the need to have interoperable healthcare systems encounters resistance from would be users. Having discussed the challenges to do with the complex nature of the healthcare domain, standardization and legal challenges arising it is important to note that these challenges play a huge role in making the users of the systems decide to remain with their systems as they are as opposed to trying to make them interoperable. Other challenges would be preference of the manual filing systems. In discussing legal challenges, the confidentiality, integrity and availability of patient information is cited as of great importance to ensure that privacy and security of the information. Breach in the security of data CIA compromises privacy of patient information that will necessitate legal suites (Iroju et al, 2013).

1.3. Problem statement

Healthcare Data Records are stored in individual healthcare data pockets that makes it impossible for that data to be accessed outside a specific healthcare facility which in turn limits the healthcare practitioners' ability to access information in a timely manner for timely and informed decision making (Iroju et al., 2013).

Assuming that a patient visits two different healthcare institutions for treatment, it is expected that each healthcare institution will have to open a file for the patient and conduct normal procedures. These procedures will be repetitive especially for chronically ill patients. To a patient, this is costly and time consuming. To a doctor who moves around the healthcare institutions, it causes inefficiency as time taken to redo the procedures slows down the entire process. For interoperability in distributed healthcare systems to be realized, the study proposes a model that enables interoperability in distributed healthcare systems. Implementation of an interoperable distributed healthcare systems will allow electronic patient data in the various healthcare institutions to be easily accessed which in turn will reduce costs to the patients (repeat medical examination) while making the practitioners more effective because of the real-time access to patient information and ease of making faster and reliable decisions.

The proposed model to handle interoperability challenges in distributed healthcare systems entirely hinges on Web Services and SOA (Hahn et al., 2010; Maciel & David, 2007). Web services is designed according to existing web protocols that are based on XML (WWW Consortium, 2012). These protocols include WSDL that describes service interfaces, SOAP that facilitates exchange between web services and client applications, and UDDI that facilitates tracking and using web services on a network (Bacon & Moody, 2002). SOA architecture is important to this study as it unifies key services of the distributed systems that characterize different functionalities, core platforms, and exchanging existing data that is in different data formats.

1.4. General objective

The main purpose of this study is to develop a model for improving interoperability of healthcare systems in a distributed environment that facilitates real-time access to patient health records.

1.5. Specific objectives

i. To establish the factors that affect interoperability in healthcare systems.

ii. To investigate interoperability Healthcare standards and architectures for a distributed healthcare environment

iii. To develop a model for enhancing interoperability in distributed healthcare environment.

iv. To test the developed model in a distributed healthcare setup.

1.6. Research Questions

i. What are the factors that affect interoperability in healthcare systems?

ii. How do the interoperability Healthcare standards and architectures affect systems in a distributed healthcare environment?

iii. How can a model be developed to enhance interoperability in distributed healthcare environment?

iv. How can the model be tested?

1.7. Justification

The benefits derived from the proposed model for improving interoperability of healthcare systems in a distributed environment include reduction in costs to patients with regard to ensuring that no repeat tests are done as the information is easily accessible, improved productivity among the doctors as the information is readily available thus helping in faster decision making and reduction in waiting time and also ensures patient health record security.

The model reduces costs by ensuring that patients' health records are accessible real-time in interoperable healthcare systems regardless of the location of the healthcare institutions. One of the greatest challenge in healthcare is when a patient is referred from one hospital to another and has to undergo a repeat of all the tests previously done at the referring hospital. A repeat of the tests means that time and money are being spent.

The proposed model aims at ensuring that doctors have real-time access to patient health records regardless of their actual location thereby allowing them to offer timely services and make informed decisions (The Regenstrief Institute, 2012). This models thus aims at making the doctors and other medical practitioners productive in their work.

The model provides a secure data access of the patient health record which seeks to guarantee confidentiality and integrity of the data being exchanged between the disparate systems. To achieve this, the platform allows different healthcare systems to allow for patients' information to be access irrespective of the platform or architecture of the system running in the various healthcare systems. This platform allows all doctors to registered institutions be able to access referred patients' information once logged in. Secure Logging in will allow for authorized personnel to access the data available to them. The platform will allow the doctors to access an entire patient's medical history thereby being able to make better decisions as all the information stored in the various hospitals can be accessed via the interoperability platform.

1.8. Scope of the Study

Interoperability in healthcare systems is aimed at allowing for information sharing among disparate or similar systems. Modelling an interoperable platform allows for doctors in different hospitals to access patient data stored in other hospitals once logged in. information obtained from Kenyatta National hospital and Mbagathi hospital are used in the development of the proposed model. The choice of this two institutions is majorly based on them being referral hospitals and their accessibility.

1.9 Ethical Consideration

The sensitivity of the data involved in this study demands the strictest code of ethics in ensuring that available data is used for all the right reasons. Patient health records cannot be made available to unauthorized users as it is highly private and confidential. The healthcare and dentistry board have to determine the suitability of this project before the actual implementation of the project.

Chapter 2 : Literature Review

2.1 Introduction

This chapter discusses theoretical frameworks that seek to help realize the importance of this study, how it will be implemented and the benefits to be derived from the development of the model. The chapter also reviews related work in bid to find better ways to develop the model.

2.2. Theoretical Framework

Theoretical framework consists of concepts, definitions and references that are relevant to this study. The framework seeks to demonstrate the understanding of selected concepts and how they relate to the focus of the study (Alabama State University, 2013).

2.2.1. Theory on Technology Adoption (TAM)

TAM is technological theory that describes how users of a system come to accept and use a given technology (Davies, 1989) and is derived from the Theory of Reasoned Action (TRA) (Ajzen and Fishbein, 1980). Factors that influence a user's behavior and decision when presented with a new technology include perceived usefulness (PU) i.e. the extent to which a user believes that by using a given system one's job performance will be enhanced and perceived ease-of-use (PEOU) i.e. the extent to which a user believes that by using a given system one would use no effort at all (Davis, 1989). TAM as a theory has been used to provide empirical evidence on existing relationships between perceived usefulness, perceived ease-of-use and use of the system (Adams, Nelson & Todd, 1992; Davis, 1989).

The study will seek to use this theory to determine the level of acceptance of the proposed model with an aim of enhancing interoperability in distributed healthcare systems. Should the users' perception on the system's usefulness, ease-of-use and its usability be highly rated then the model will have been accepted else the model will have been denied meaning that a given healthcare institution will not adopt it for its day-to-day operations.

The Limitation of TAM is that in determining the adoption of technology based on user intentions, the theory to a larger extent is biased as a user's intentions are varied and have different causatives (Bagozzi, 2007). It is good to not that intentions can be tagged to a user's lack of technological knowhow, attitude towards technology and organizational politics among others.

2.2.2 Organization Theory

This is the study of the designs, structures and organizational relationships within organizations and the behavior of an organization's users. Organization theory can be viewed based on normative, historical, empirical, social and international institutionalism approaches (Peters, 2000).

Normative approach argues that one acquires certain values based on interacting with a given institution i.e. one's behavior is derived from the organizational normative standards as opposed to maximizing individual benefits or values (March and Olsen, 1984; March et al, 1989; and March et al, 1996). The historical approach looks at the policies and structural decisions made at the beginning of the institution as having influence on how the institution will transact or conduct itself throughout its life (Steinmo, Thelen and Longstreth, 1992). The importance of this approach is that it defines to a greater depth the ability with which to effect changes in an institution's policies and structures. Empirical approach reviews the impact of policy change to the overall governmental / organizational structure (Weaver and Rockman, 1993; Von, 1996) that is, change in a policy impacts heavily on the organization hence pros and cons must be put up for consideration before effecting or declining them.

Organization Theory has been used to propose ways in which an organization can manage rapid changes taking place so as to maximize on its benefits while also ensuring that the rapid changes do not negatively affect the organization. Interoperability in distributed healthcare systems alters how one organization (Distributed system in healthcare institution A) relates or works with another organization (Distributed system in healthcare institution B, C…) to ensure that its objectives are met.

2.2.3 Theory of Expectancy

The goal of this theory is to motivate the individual i.e. it assumes that behavior is derived from a certain level of awareness when choosing from among options. The theory submits that individuals can be inspired if they consider that performance is tagged on effort, good reward, meets ones' needs and ensures that the effort is worthwhile (Vroom & Deci, 1983). It is based on valence which describes the value one places on the outcomes of a given process such that the value is either extrinsic (external satisfaction / benefits) or intrinsic (internal satisfaction) or both.

Expectancy describes levels of expectations on one's capabilities. Those capabilities can be enhanced by making required resources available at all times and properly training those charged with managing and using the resources. Instrumentality describes the driving force behind a fulfilled individual as being reward (value), a reward that must be delivered. The implementation and adoption of a system based on this theory is pegged on the value one places on the system, the ease of use of the system based on capabilities and the reward be it intrinsic or extrinsic that comes with its use.

2.3 Distributed Health Systems

A distributed system can be described as a collection of autonomous computer systems that are viewed by the users as a single system (Tannenbaum & Maarten, 2007). For healthcare systems, this would mean a set of independent healthcare systems from different vendors and institutions that are integrated and appear to the users as a single system. Based on the description provided, the following emerge as the key characteristics of a distributed system: the users are not aware of the variations in the systems and how they communicate; users and applications interaction with the distributed system must be consistent and unified irrespective of the point of interaction; and expandability or scalability must be achievable.

2.3.1 Objective of a Distributed Healthcare System

The main objective of a distributed healthcare system is to ensure easy access to available resources, transparency (a distributed system viewed as a single system), openness of the distributed system that meets clearly defined standard rules (semantics and syntax describing each services) and scalability (Tanenbaum & Maarten, 2007). For a healthcare distributed system to be fully realized, there is need for the standardization of the syntactic and semantic. This standardization aims at having different systems communicate without entirely having to do major configurations or changes.

2.4. Factors that influence healthcare systems interoperability in a distributed environment

The need for an interoperability system in a distributed environment is of great importance in that a series of variables are taken care of. These variables include ease of access by authorized persons, data portability, data confidentiality, integrity and security, capture of different data formats, sharing files, reduction in costs, and scalability of systems (Ministry of Health, 2010). These factors as identified by Ministry of Health (2010) are discussed below:

2.4.1 Ease of Access

The need for an integrated system is so that the HIS may be accessible by authorized persons who perform authorized operations on the data. The doctor needs access to the system to update a patient's medical record, a pharmacist accesses the system to update the prescription and payment and a patient access the system for self-management. This accessibility must be made possible while ensuring security is assured. Access should also be defined in the context of either being offline or online. Is the system accessible only internal to an organization or external, from anywhere? A couple of security challenges arise.

2.4.2 Data Portability

Different systems should be able to allow availability of data regardless of the platform the systems use. Data in one system should be accessible to another and be in a usable format to provide necessary information for ease of decision making.

2.4.3 Data Confidentiality, Integrity and Security

Confidentiality of data seeks to guarantee privacy. Confidential data strictly demands for authorized access. Data integrity on the other hand ensures reliability of the data. Authorized updates are only made to the available data and that database state is kept consistent throughout. Data security defines the levels of authorization and authentication done for any updates performed on the data. The three properties of data are to ensure that the healthcare system data is not compromised.

2.4.4 Capture of different data formats

Data capture in healthcare systems can be in varied formats such as texts, images, videos and emails. These formats need a certain level of standardization to ensure that it is portable in the different systems existing.

2.4.5 Sharing files

Integration of the healthcare systems necessitates access to shared files on and off the network making it possible to perform patient and disease management regardless of the physical location. This makes the data available and allows for faster decision making that leads to efficiency and productivity.

2.4.6 Reduction in costs

An integrated and interoperable system reduces cost of services both to the institution and to the patient in that records are easily accessible thus reducing repetitive and / administrative overheads that existed. Rather than do repeated procedures for a patient whose file cannot be found, an integrated healthcare system would make resources available leading to efficiency and cost reductions.

2.4.7 Scalability

As operations increase in an organization, needs increase and at times change, some of the needs demand for a robust system that the legacy system cannot manage. Because of growth need, the systems need be scaled which means that new systems might be acquired which differ with the current system leading to the need of integration so as to be able to access the data.

2.5. Analysis of Interoperability Approaches within a Distributed Healthcare Environment

When interconnecting different systems, coming up with set standards would help ensure that the systems adhere to set standards hence interoperability is easily achievable. One model that employs the use of standards to ensure interoperability of systems is the Open Systems Interconnect (OSI) model. Standards in a distributed healthcare environment include semantic and syntactic operability. Other than standards, other approaches used are as described below (Tanenbaum & Maarten, 2007):

Frameworks that ensure interoperability by setting up specifications and policies to be used between its agencies and service delivery to the public. Specifications and policies need reflect the current technology. The frameworks consist of specifications and policies covering interconnectivity, data integration, e-service access and content management. Enterprise architecture aims at aligning the IT solution with the enterprise business processes and goals. Navigator provides a framework that helps organizations improve their sub-systems towards attaining distributed systems interoperability. Web services are self-contained and modular applications that can be described, published, located and invoked over the web. The development of middleware software is an approach used to ensure interconnectivity thereby achieving interoperability. The middleware provides a platform that interconnects all systems regardless of the design and platform and thus standards for developing the middleware application required.

2.6 Interoperability Healthcare Standards and Architectures for a Distributed Healthcare Environment

Standards can be described as a set of rules and procedures that govern formatting, content, and significance of sent and received messages (Tannenbaum & Maarten, 2007). The study sought to look at available healthcare standards in a distributed environment and see how they can be applied. Some of the advantages of having acceptable standards for healthcare processes (ESHI, 2000) include competition and reduction in costs; ease of replacing or updating standardized systems; ease of communication between heterogeneous systems that allows compatibility in data exchange (HIMSS, 2006), ability of organizations to repeatedly extend their services / capabilities; and ensure reduction in errors be they system or data errors.

There exist several organizations that provide electronic health records standards that manage distributed healthcare systems for purposes of standardized implementation, structuring, integration, sharing and interoperability in distributed healthcare systems. Some of these existing standards include ISO (International Organization for Standardization, 2012), CFR (The World Health Organization, 2012), ASTM (ASTM

International, 2012), National Electrical Manufacturers Association (NEMA, 2012), HL7 (The Health Level Seven, 2012), CEN (The European Committee for Standardization, 2012), and ONCHIT (US Department of Health and Human and Services, 2012). Besides standards managing electronic health records, coding systems are critical to the development of an interoperable distributed healthcare systems for connecting heterogeneous systems having different terminologies. Encoding standards have been provided for by organizations such as AMA (The American Medical Association, 2012), Regenstrief (The Regenstrief Institute, 2012), CMS (US Department of Health & Human and Services, 2012), IHTSDO (International Health Terminology Standards Development Organization, 2012), and WHO (The World Health Organization, 2012). These organizations provide standards for encoding healthcare data and knowledge.

Standards used in distributed healthcare systems can be categorized into two main categories that is syntactic standards for communication or information exchange and semantic standards for documentation purposes (Sunyaev et al., 2008).

2.6.1 Syntactic Standards

Syntactic standards seek to ensure accurate transmission of medical and administrative data between heterogeneous and distributed systems (Schweiger et al.,2007). The study differentiates that data i.e. medical data that is used in a patient's treatment process and administrative data that is used for basic administration in the healthcare institution.

Established standards (Pedersen & Hasselbring, 2004) used for syntactic standardization include: Health Level 7/Clinical Document Architecture, (HL7/CDA); Digital Imaging and Communications in Medicine (DICOM); and Electronic Data Interchange for Administration, Commerce and Transport (EDIFACT). These standards are widely accepted mainly because of their openness and other factors as listed below:

i. Vendor-independence

ii. Main standard for information exchange between systems.

iii. Functions at the application layer of the OSI reference model.

iv. CDA describes the structure and content of healthcare documents based on XML format.

v. Information use can either be individually or integrated.

vi. Open standard.

vii. Used for exchange or communication of images.

viii. Store images in TIFF and JPEG formats.

ix. Facilitates electronic archiving.

x. Standardize formats for electronic communication or exchange of administrative data.

2.6.2 Semantic Standards

All communicated, transmitted or exchanged data between the heterogeneous systems require accurate interpretation for correct or accurate diagnosis and medical prescription. For this to take place, the semantic standards are required for the encoding of healthcare data, (Sunyaev et al., 2008). Standards are derived either via classification or terminology system. factors for consideration in the standardization of the semantic standards are as below:

i. Classify illnesses and health related issues.

ii. Combine terms derived from determined concept orders.

iii. Cover the arrangement of a unified terminology for expressions in the medical field and offer a multidimensional terminology system for healthcare laboratories.

iv. Facilitate complete description of all medical conditions and extension of nomenclature.

v. Incorporates key medical terms into one and represent all available term relations.

2.6.3 Component of interoperability standards for a distributed health environment

Healthcare terminologies – syntax and semantic

These are the terms used in the development of the system and documentation in healthcare respectively. Syntax defines how the terms used in programming will be structured to ensure that there is commonality in how the systems are able to interact. It is good to note that terminologies differ from one nation or continent hence the need to standardize. Semantic defines the terminologies used in documenting the system for use. This allows for ease of referencing in the event of any access problem. It can also act as a guide for those with limited healthcare knowledge.

Healthcare message transmission

Data format should be in a standard that allows another system to access the data and be able to interpret it. These formats range from texts, emails, images and audios / videos. The establishment of standards for data formats allow for faster and quicker data transfer and sharing.

Healthcare ontologies

The relationships between data entities need to be established so to manage the content and the structure of the entities. This is one of the key components in arriving at interoperability in distributed healthcare systems.

Privacy and security

For interoperability to be fully achieved, issues of who has access to the system (authentication) and who has access to specific areas of the system (authorization) caters for security which if implemented very well enhances privacy. The sensitivity of the patient health records demands the highest level of privacy and security.

Network

Data accessibility in interconnected systems is over the network and therefore need for a secured LAN, WAN and Wireless network is of great importance. Other factors that come with network is bandwidth to manage the transfer of different types of data transmitted over the network.

Platform independent

The ability of the system to run on any platform such us Linux, Windows, Mac OS and android Operating System is important to achieving interoperability in interconnected systems. Systems able to run on all of these operating systems require specific standards to ensure they operate as expected via web interface.

2.6.4 Models, architectures and technology used in distributed systems

When designing distributed systems, it is expected that the design finds a balance between systems performance, its dependability, ease of managing of the system, and security. In choosing an architectural style used during distributed systems design, care should be taken to ensure that a system's key non-functional requirements are fully supported. This research discusses five architectures:

i. Master-slave architecture is used in real-time systems where definite interaction response time is necessary.

ii. Two-tier client–server architecture is used for basic client–server systems, and centralization of the system purely on the basis of security which is arrived at through encrypting the communication.

iii. Multitier client–server architecture is used when the volume of transactions the server needs to process is high.

iv. Distributed component architecture is used when different databases and systems resources need to be combined, or as a model that implements multi-tier client–server systems.

v. Peer-to-peer architecture is used when information stored locally is exchanged by clients while the server introduces the clients to each other. It may also be implemented when independent computations are made in large number.

2.6.4.1 Master-slave architectures

This architecture is utilized in real-time systems where separate processors associated with the acquisition of data from the environment the system exists in, processing of the data, and computation and management of the actuator. Actuators are devices that are controlled by the system software whose actions change a system's environment. The responsibility of the master process is to compute, coordinate, communicate and control slave process since slave process dedicate themselves to specific actions (Erl, 2005).

You use this master-slave model of a distributed system is used in:

i. Predicting the required distributed processing and confining processing to slave processors.

ii. Meeting set processing deadlines.

iii. Computationally intensive processing by slave processors.

2.6.4.2 Two - tier client–server architectures

In a client–server systems, the client is the user's computer from which the application system runs whereas the server is the remote computer that hosts the system (Erl, 2005). This architecture is implemented as a single logical server with unlimited number of clients accessing the server:

i. A thin-client model, implements the presentation layer on the client machine whereas all other layers are implemented on the server. It is simple to manage the clients.

ii. A fat-client model, allows all or part of application processing to be carried out on the client whereas database functions and data management are implemented on the server.

The advantage thin-client model has is that its clients' management process is simple. Increased number of clients makes new software installation to be expensive and difficult and to install new software on all of them. Use of web browsers dispels the need for software installation. The thin-client model's disadvantage is that it burdens the server and the network with heavy load that needs to be processed. With this in mind, its implementation thus requires extra investment in server and network capacity.

The fat-client model uses resources available on the host computer and allocates presentation and application processing to the client. A challenge that arises is the need for extra system management and software maintenance required on the client machine. Although fat-client's processing distribution is more effective than thin-client's, it has a complex system management since functionality application is accessible by many computers. This means that a change in the application will demand reinstallation on all clients leading to extra costs. An example of a fat-client model is ATM used in banks (Erl, 2005).

2.6.4.3 Multi-tier client–server architectures

The demand to map a system's logical layers i.e. application processing, presentation, database, and data management on to the client and the server systems poses as the main challenge for the two-tier client–server model (Erl, 2005). These leads to scalability, system management and performance problems. Multi-tier client–server architecture provides a solution to these challenges by separating the logical layers so that each executes on a different processor.

The three tiers coordinate in a way that:

i. Database services are provided by the customer database.
ii. Data management services are provided by a web server.
iii. The web server implements application services in the form of scripts that the client executes.

This architecture enables optimized transfer of information between the database and the web servers. Information exchange between systems use low-level and fast data exchange protocols. Retrieving information from the database is handled by an efficient middleware that manages Structured Query Language (SQL) database queries. A multi-tier variant architecture is a three-tier client–server model whose system contains additional servers (Erl, 2005).

Application processing being the most volatile aspect of the system demands that this process be distributed across several servers making multi-tier client–server more scalable as compared to two-tier model and also owing to its centralized location, it can easily be updated.

2.6.4.4 Distributed component architectures

The layered approach organizes processes into layers such that each layer is implemented as a distinct logical server. The limitations to this approach is its lack of design flexibility that should be done for each layer and the demand to plan for its scalability to accommodate more clients. Distributed component architecture structures the system as a set of interrelating components or objects that make available an interface to a series of available services. These services are available to other components via middleware that is facilitated by method or remote procedure calls. Distributed component architecture relies on middleware which manages object interactions, resolves variances between parameters types handled between object, and provides a series of shared services that application object use. Some of the existing middleware include CORBA (Orfali et al., 1997), .NET and Enterprise Java Beans (EJB).

The following are the advantages of using a distributed component model:

i. It lets the designers to delay decisions on how and where services ought to be provided. The components that provide services can execute on any node on the network.

ii. Being an open system architecture, it allows for required resources to be made available if and when needed without causing major disruptions on the existing system.

iii. It is flexible and scalable i.e. new components can be added with increase in the system load without halting other system services.

iv. The system can be reconfigured dynamically as the components or objects move across the network as expected. This is an important aspect especially where demand on services keeps fluctuating. To improve a system's performance, a component that provides a service can move to the same processor as the service that is requesting components.

This architecture can be implemented as a logical model that allows one to organize and structure the entire system (Erl, 2005). This can be achieved by providing the functionality of an application either as a service or a combination of services which can be provided by a set of distributed objects. For example, in a healthcare application there may be application objects dealing with patient management, pharmacy, and imaging among others. This architecture would best illustrated in data mining systems which look for data associations stored in a set of databases. Data associations are arrived at by separating the databases, conducting intense process computations and graphically representing the results (Erl, 2005).

The advantage of this architecture over the layered one is that there is minimal disruption with the addition of new databases as each added database is made accessible by adding another component which simplifies the interfaces that control data access. These databases may be hosted on different hosts. Having new integrator objects allows for the mining of new forms of relationship.

There are two major disadvantages of this architecture that include its design complexity when compared to the client–server model that makes it hard for one to envision and comprehend and the lack of acceptance of standardized middleware by the users arising from its complexity. Although service-oriented architectures offer solutions to these problems, distributed component architectures perform better and have high throughput since message-based interaction are slower than RPC communications.

2.6.4.5 Peer-to-peer (p2p) architectures

The distinction made between servers in the client–server architecture causes an undistributed load in the system so that servers get overworked. This leads in increased spending on the servers while the clients' processing capacity remain underutilized.

Peer-to-peer systems are systems that are decentralized and whose computations on the network may be facilitated by any node (Erl, 2005). This is achievable on the merit that there is no differentiating between the clients and servers. This allows the overall system to benefit from the optimized performance and available storage across a pull of computer resources. Communications across the nodes are enabled by standards and protocols that are entrenched in the applications and each node is made to execute a copy of the application. It is worth noting that this architecture is preferably used for individual rather than commercial systems (Oram, 2001). Business that explore this architecture, do so to maximize their computer networks potential (McDougall, 2000).

It is appropriate to use a peer-to-peer architectural model for a system in two circumstances:

This architecture is used where the system's performance is intensive yet it allows the required processing to be separated into large sets of autonomous computations and where the system's main objective is to exchange decentralized information between autonomous computers on the network that do not require management. In a decentralized peer-to-peer system, the network nodes not only functional elements but also act as communications channels that manage routing of data and controlling signals from one node to the next. The advantage of this decentralized system is that it enhances redundancy in the event that both tolerant and fault-tolerant nodes disconnect. Its disadvantages are that the different nodes on the network may perform a similar search leading to increased overheads in repeated peer communications (Erl, 2005).

Semi-centralized architecture reduces traffic existing in the nodes by ensuring that communication between nodes is facilitated by one or more nodes acting as servers. The role of the server is either to facilitate communication between network peers or manage computation results. Whereas this architecture allows for efficient maximization of network capacity, its usage has been inhibited by its lack of trust and security. Since access is open to peers, there is unrestricted access to one's resources which brings about insecurity.

2.6.4.6 Service Oriented architecture

Service-oriented architectures (SOAs) are a means by which distributed systems are developed such that the components of a system are stand-alone services and are implemented on geographically distributed systems. SOAP and WSDL are XML-based protocols that support both exchange of information and service communication (Erl, 2005). XML is a notation readable by both human and machines i.e. it permits the definition of structured data such that text is marked with a meaningful and unique identifier. Consequently, services are autonomous to the platform and language used to implement it. To develop a software system, combine both local and external services from the various providers by ensuring there exists seamless communication between the system services. Because of SOA standardization, the architecture is not prone to incompatibilities that come with changes in technology brought about by the various suppliers or developers.

Technologies used in XML and their roles in web services (Erl, 2004) include:

i. SOAP is a standard that allows for exchanging messages hence supporting the existence of communication between services. It achieves this by defining the optional and critical components of messages that are exchanged between services.

ii. The Web Service Definition Language (WSDL) is a standard for defining service interface. It clearly marks out how the service operations and bindings ought to be defined.

iii. WS-BPEL is a workflow language standard that defines the process programs that involve a number of different services.

Developing service-based applications enables companies to collaborate and utilize each other's organizational functions. This allows for automation of systems having extensive exchange of information across organization boundaries. These applications are achieved by linking various providers' services using specialized workflow language or standard programming language.

Below is a summary of the Service-Oriented Approaches (Newcomer & Lomow, 2005):

"Driven by the convergence of key technologies and the universal adoption of Web services, the service-oriented enterprise promises to significantly improve corporate agility, speed time-to-market for new products and services, reduce IT costs and improve operational efficiency."

Services as reusable components

A service can therefore be defined as the following:

A loosely-coupled, reusable software component that encapsulates discrete functionality, which may be distributed and programmatically accessed. A web service is a service that is accessed using standard Internet and XML based protocols… (Councill & Heineman, 2001). Services communication entails exchange of messages executed in XML. The distribution of these messages is by use of Internet transport protocols. Services do not use method calls or RPC to access certain functions associated with other services unlike software components. The major problem with WSDL is that the definition of the service interface does not include any information about the semantics of the service or its non-functional characteristics, such as performance and dependability.

Service Engineering

In service engineering, reusable services are developed in service-oriented applications. Each service must be a representation of an abstraction that is reusable and useful to different systems. There is need to design and develop functionalities that are useful which can be link with abstraction while ensuring the robustness and reliability of the service. Service engineering process has three logical stages namely service candidate identification, service design and service implementation and deployment. Service candidate identification, identifies services to be implemented after which it define the requirements the service needs. Service design designs both WSDL and logical service interfaces. Service implementation and deployment manages implementation and testing of the service before deployment or making it available to users.

Service Candidate Identification

Service candidate identification involves understanding and analyzing the organization's business processes to decide which reusable services could be implemented to support these processes. The identified services include utility, business and process or coordination services. Utility service deal with general functionality, business service engages a specific function and coordination service supports general business process. Services can be said to be entity-oriented or task-oriented (Erl, 2005). Entity-oriented services are like components whereas task-oriented services are linked with some activity.

Service Interface Design

This defines the processes related to parameters and service. There must be careful consideration taken regarding service operations design and messages. Precedence must be given to the minimization of the number of exchange message taking place so as to complete the requested service. The design process might entail logical interface design, message design and WSDL development

Service Implementation and Deployment

A service can be developed and executed by programming service interfaces to existing objects using standard programming language. Deployment of a system is dependent on the development and testing phase so that once deployed, the system should be ready for use.

Legacy System Services

Due to the cost implication that comes with rewriting or replacing an entire existing system, obsolete technology has always been used in organizations hence the term legacy systems. Using services makes it easy to implement wrappers that provides access to a legacy system's data and process, this enhances integration.

Workflow Design and Implementation

This is aimed at analyzing a given business function to determine the various activities that are carried out and how those functions exchange information.

Service Testing

System testing helps determine whether a given system addresses its functional and non-functional requirements. Testing also identifies errors or flaws that arise during the development stage of the system. For a thorough testing to be carried out source code analysis is critical as it makes it easy to identify errors. In summary, having discusses the thin and fat client-server architecture, multi client-server architecture, distributed component architecture, peer-to-peer architecture and service oriented architecture, the merits of service oriented architecture make it possible to explore interoperability in distributed systems with ease.

2.7 An API-based Model for Improving Interoperability in Distributed Healthcare Environment

Application programming interface (API) specifies how software applications interact with each other regardless of the platform where these applications are residing (Dana & Ciprian, 2014). APIs' allow different applications to request information from each other and also use each other facilities. This can be implemented using Web-based technologies such as SOAP, REST based services or higher-level programming languages.

The two main API categories are in-process API and remote API. In-process API combines objects, methods, functions, or procedures while abstracting resources including memory usage, mutable of immutable, usually transparent, data-structures and usually opaque pieces of machine executable or interpreted code. Remote API on the other side bridges applications in the form of Web services, remote calls, message passing, or application dependent protocols. With regard to usability, remote APIs are harder to implement that in-process API.

The three main components involved in API design include the application, API code, and the client / host (Geert, 2017) as captured in Figure 2.1 below:

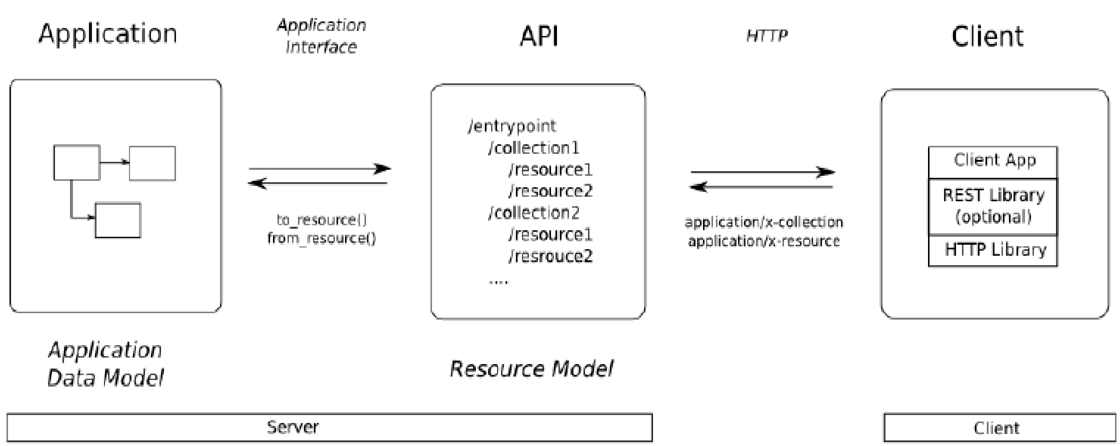

Figure 2.1:API Components (Adapted from Geert, 2017).

2.7.1 Application

The application exists independent of the existing API (Geert, 2017). This essentially means that you can reprogram the Application without necessarily having to alter anything on the API. This independence allows for increased automation and efficient system development while at the same time reducing the limitations that come with a dependent system.

2.7.2 API

The API accesses both the application and operations state through the application interface, and represents it as an API. An API resource model is one that allows for the transformation, operations and relationships existing between the resources. Resources are the foundation in API. A resource is an object with a type, associated data, relationships to other resources, and a set of methods that operate on it (Geert, 2017).

2.7.3 Client

According to Geert (2017) at one point of the operations between the application and the client, either can take on the role of client or host.

2.8 Conceptual model of this study

The proposed system as seen in Figure 2.1 has a knowledge base with patients' data from where the platform for centrally accessing data from either system is accessed.

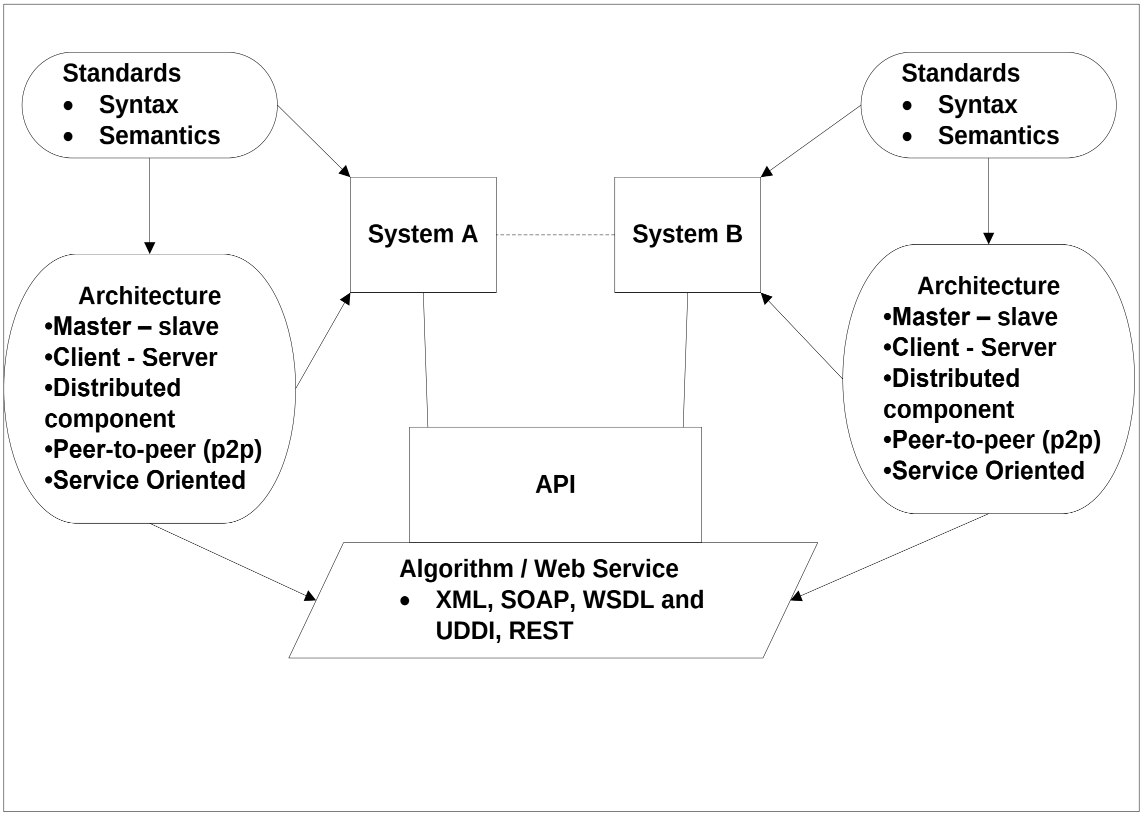

Figure 2.2: Conceptual Diagram of the interoperable model between Distributed Healthcare Systems.

Chapter 3 : Research Methodology

3.1 Introduction

The research is aimed at finding a way of implementing interoperability in healthcare systems that have different platforms or architectures and allow for data to be shared in the different systems. This chapter describes the methods used to conduct this research and its viability, the target population, sample size, data collection and analysis procedures. This chapter further discusses system analysis approaches, architectures, design, development, implementation and testing.

3.2 Agile Software Development Methodology

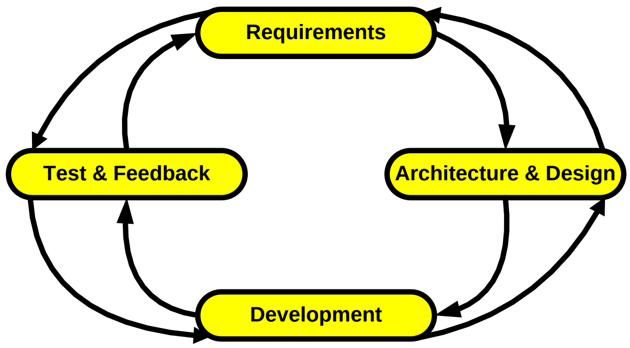

Figure 3.1: Agile Software Development (Adapted from CPrime, 2014).

Agile methodology was applied in this research to enable faster iteration process defined by frequent releases that capture detailed user feedback. This frequent releases doubled up with detailed user feedback allow for detailed and organized improvements to be effected (CPrime, 2014). The iterative processes used in Agile methodology is captured in Figure 3.1. above.

Figure 3.1 provides a detailed step by step approach used in the successful implementation of the research objectives. The requirements step was used to gather required information on the systems from which the interoperability of the two systems was derived. The architecture and design step reviewed the architectures of the identified systems and after that designed the API used to allow data in the disparate systems to communicate. Development step allowed for the actual development of the API to allow data interchange between the systems. The last and final step was test and feedback which ensured that the

API was highly developed capturing all requirements. The tests done were mainly based on the ease of the different system accessing the necessary data if and when called upon.

3.2.1 System Architecture

Representational State Transfer (REST) architecture (Roy, 2000) is to be used in the development of the API which incorporates six constraints that include Uniform Interface, Stateless, Cacheable, Client-Server, Layered System and Code on Demand. Uniform interface outlines the interface existing between the client and the server. Stateless outlines that a request contains the necessary state to manage itself. Cache allows for client responses to be cached. Client-Server brings about independence as client interface does not concern itself with the server interface and vice versa. One can be altered without affecting the either. Layered system allows the application to either exist on an intermediary server or the end server or both with brings about issues of scalability. The study adopted the Client-Server architecture.

3.2.2 System Analysis and Design

Data, process and object oriented approaches are three key approaches in the development of any system. The research used the data-oriented approach as this is what the system requires to access from the knowledge base and provide it to the various users. It must be noted that Data-oriented approach cannot be holistically used by itself hence the need to use Object-Oriented approach.

3.2.3 System implementation

A model of the existing systems in KNH and Mbagathi hospital was developed using PHP and MySQL. PHP as a tool is most preferred as it is not dependent on any platform, (PHP, 2017).

3.2.4 System Testing

Usability testing was carried to determine if the functions of the model allow for interoperability of the healthcare systems. The objective was to allow for information in 2 disparate systems to be shared across.

3.3 Research Design

The research used qualitative and quantitative as its research methodologies. The objective of using qualitative method was to provide clear understanding of the current and existing systems and platforms. The quantitative method on the other and was used determine the adoptability and usability of the system (California State University, 2012).

3.4 Target population

The research will be carried between two hospitals, Kenyatta National Hospital and Mbagathi Hospital. The target population, users of information in a referral hospital and referring hospital which includes users of the healthcare management system e.g. doctors and patients whose age is above 18 years.

3.5 Sample techniques and sample sizes

Target population refers to a representation of the whole population studied to capture information about the entire population of study (Kothari, 2004). The study covers referral and national hospital that is Mbagathi and Kenyatta National hospital respectively. These organizations were selected on the merit that interoperability in distributed healthcare systems would bring to their operations and services. With the intention of interviewing specific actors in the system that is doctors, patients and system administrators. This brought the total population size to 6000. The focus on this area is as a result of the available systems in use with a need to check their interconnectivity. The criteria for deriving the sample size is based on precision level, confidence / risk level and the extent of variability in features being quantified. 95% level of confidence and 1.96% error in sampling is used in determining the sample size.

Sampling Formula

$$n = \frac{NZ^2 * 0.25}{[d^2 * (N-1)] + (Z^2 * 0.25)}$$

n = Sample size

N = population

d = level of precision (0.10 – 0.05)

Z = level of confidence that is 1.96 for 95% confidence level

$$n = \frac{6000*1.96^2*0.25}{[0.1^2*(6000-1)] + (1.96^2*0.25)}$$

$n = 94.5 \equiv 95$. The sample population or size is thus set to 95.

3.6 Data Collection Procedures

Data sources can be said to fall into two main categories i.e. primary and secondary (Bickman & Rog, 2008). Primary data sources are obtained from people, observation of events, documentation reviews and test data. Secondary sources on the other hand include existing healthcare systems, administrative records and various research findings related to this research. Primary data obtained using interviews and questionnaires will seek immediate information from the selected respondents while secondary data will prior be collected which will be statistically processed (Kothari, 2004). Primary data was sought using interviews and questionnaires sent to all the respondents. Secondary data included documentation evidence which allowed the researcher to review the existing systems for interoperability purposes. Secondary data included system's documentation from the research population and internet sources from where published research work related to this research was obtained.

3.6.1 Questionnaire

This is a research tool used to gather information from the research population using a list of questions to be answered by the respondents. Each question in the questionnaire is aimed at addressing a specific objective, research question or hypothesis of the research (Mugenda & Mugenda, 2003). The research questionnaire had a mixture of both structured and unstructured questions. Structure questions aim at getting specific information whereas unstructured questions allowed the respondents to provide more information they deemed fit for use in this research as highlighted in.

The questionnaire used to conduct the interviews were printed and provided in hardcopy to the respondents. This was convenient as it allowed the researcher to have first-hand experience with the existing healthcare systems or technologies and helps get more information that would not have been possible were it to be online.

3.7 Data analysis procedures

Data analysis as a research process accesses the research's primary data for purposes of relating the data to identified problems. Data analysis examines the data collected from the interview and questionnaires so as to make deductions which help acquire meaning, draft conclusions and make certain assertions important to this research (Kombo & Tromp, 2006). The study used descriptive statistics to analyze the quantitative data. After summarizing and tabulating the data using Microsoft Excel 2013 they were presented in tables and percentages making it possible to derive their meaning.

3.8 Research Quality

Data collected in this research was examined for its entirety, unambiguousness, dependability and reliability such that errors and biases that could easily have arisen were avoided.

3.8.1 Ethical Issues in Research

In adhering to the moral and legal requirements of research, the researcher ensured that access to required data, its privacy and confidentiality, and its protection and storage was purely limited to the research.

Access to required data: Authorization was sought and obtained from the various institutions to allow for the interviews to be conducted within the various functional areas. The data used in the research was made available by the full knowledge of the selected institutions. Privacy and confidentiality: The research sought to ensure that data obtained for the study was strictly used for the purposes of this study and that in no way was it made available in the public domain. Data protection and storage: The obtained data was stored in a secure location only accessible to the researcher and the supervisor upon demand. This means that access was limited only to authorized persons.

3.8.2 Reliability

Reliability is determined by the consistency levels of the results over a specified period of time the ability to correctly illustrate the aggregate population used in a research study. A research therefore is considered reliable in the event that its outcome can be replicated in similar approaches (Golafshani, 2003). In this research reliability was achieved by issuing respondents questionnaires to fill after which a correlation of the data obtained from the 2 hospitals checked. This facilitated the researcher to proceed with the study.

3.8.3 Validity

A dependable research design maximizes validity by providing a clear explanation of the object of study in research and controlling likely biases that could misrepresent the research findings (Bickman, 1989). There exist four types of validity which are internal validity that determines the extent of drawing causative conclusions, external validity determines the extent of data generalization, statistical conclusion validity determines the appropriateness of the statistical methods used and their desired effects, and construct validity which determines constructs implemented successfully in the conceptual framework. In respect to the study, the emphasis was placed on construct validity.

Chapter 4 : System Design and Architecture

4.1 Introduction

The main purpose of this study is to come up with an effective and efficient way of allowing referral hospitals to access patient health records from referring healthcare institutions with the aim of attending to the patiently promptly. Object Oriented Analysis and Design was used in this research.

This chapter contains data analysis carried out on the interviews carried out on the sample population, system analysis and design. The system is analyzed based on the collected data after which the system design is implemented.

4.2 Results from Questionnaire

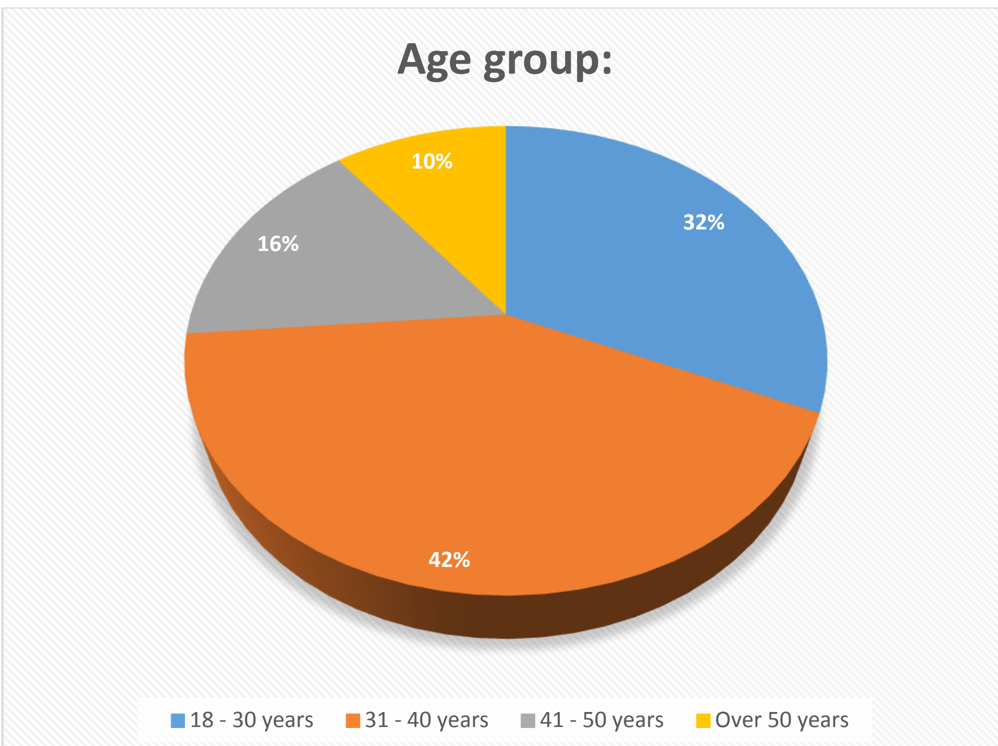

Figure 4.1: Age group

From Figure 4.1, 26% of those interviewed are 41 years, 32% caters for those aged between 31 and 40 while 42% caters for 42% below. This is important in determining the ease of use, level of acceptance and usability of the system.

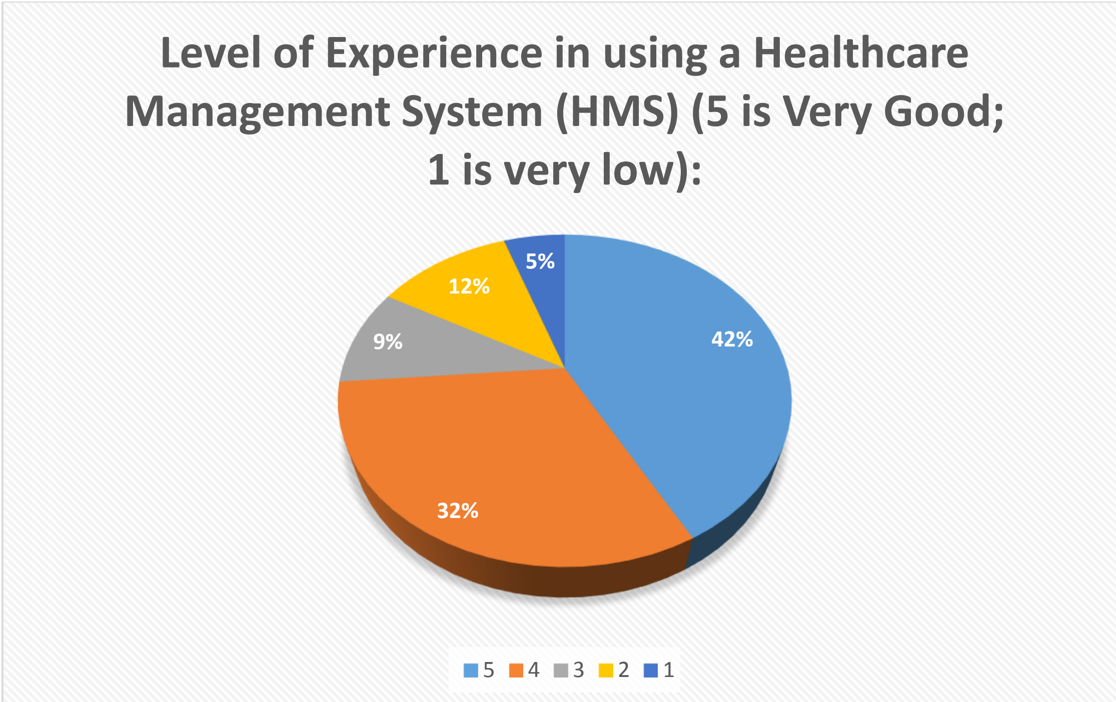

Figure 4.2: Rate of User experience using HMS

From Figure 4.2, 83% rate themselves to have good experience in using Healthcare Management System. This is important in the sense that adopting to use the API would be readily acceptable.

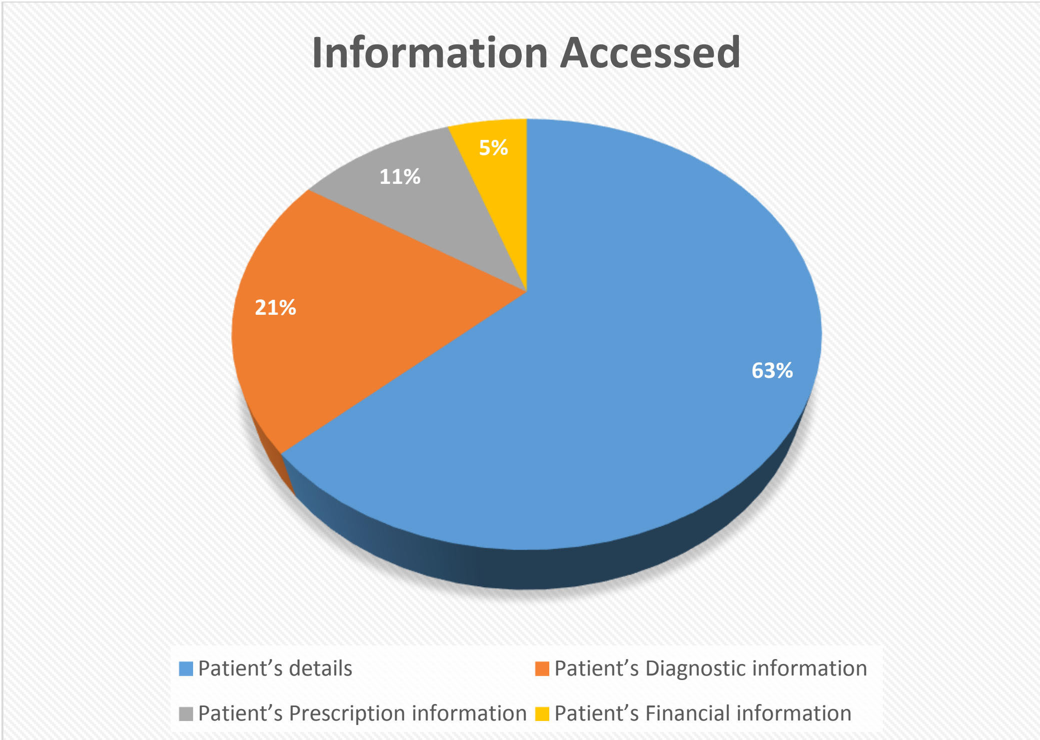

Figure 4.3: Information accessed

From Figure 4.3, The main reason for accessing patient health record and as such, 84% of the population access the information with the intent to treat patients i.e. 63% access patients' record while 21% caters for patients' diagnostics. Other feedback obtained on the same include:

Importance of information obtained from HMS can be summarized as:

i. Efficiency in making decisions.

ii. Effectiveness at work as little time is waste looking for physical files and doing manual write-ups.

iii. Increased productivity as one is able to do more than without the system.

iv. Allows for quick referencing hence boosting confidence both to the doctor and patient.

v. Ease of recording and retrieval

Information management challenges while attending to referral patients:

i. Lack of patient health record from the referring institution slows down the entire process.

ii. Makes work repetitive i.e. a patient is take through procedures that he or she may already have finished.

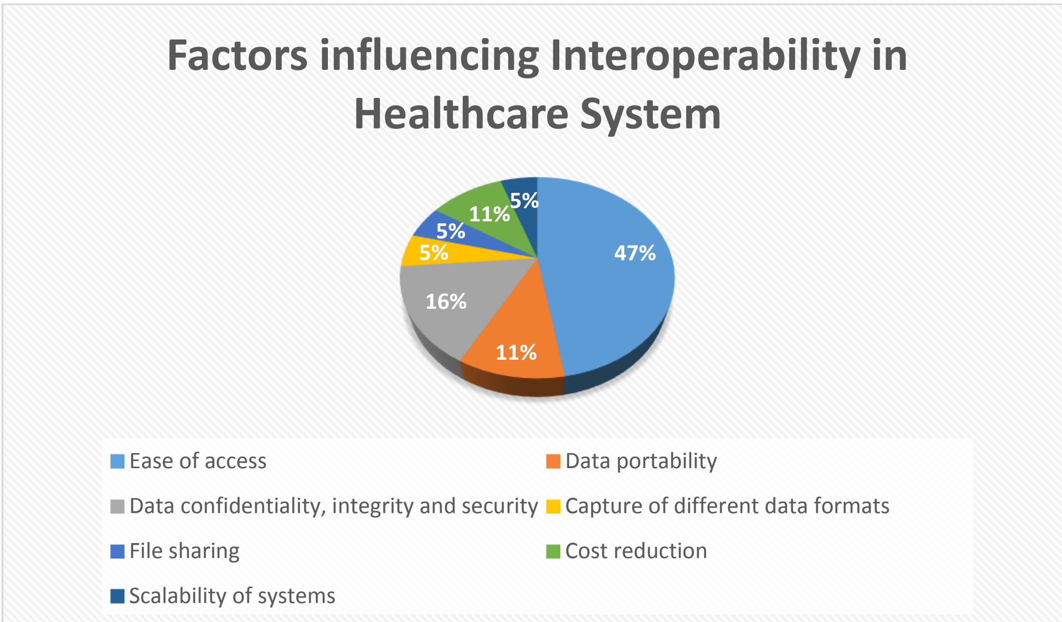

Figure 4.4: Factors influencing interoperability

Figure 4.4 above captures the factors that influence interoperability in healthcare systems and the degree of importance for each. Ease of access is sited as the most significant factor at 47% followed by data confidentiality, integrity and security at 16%.

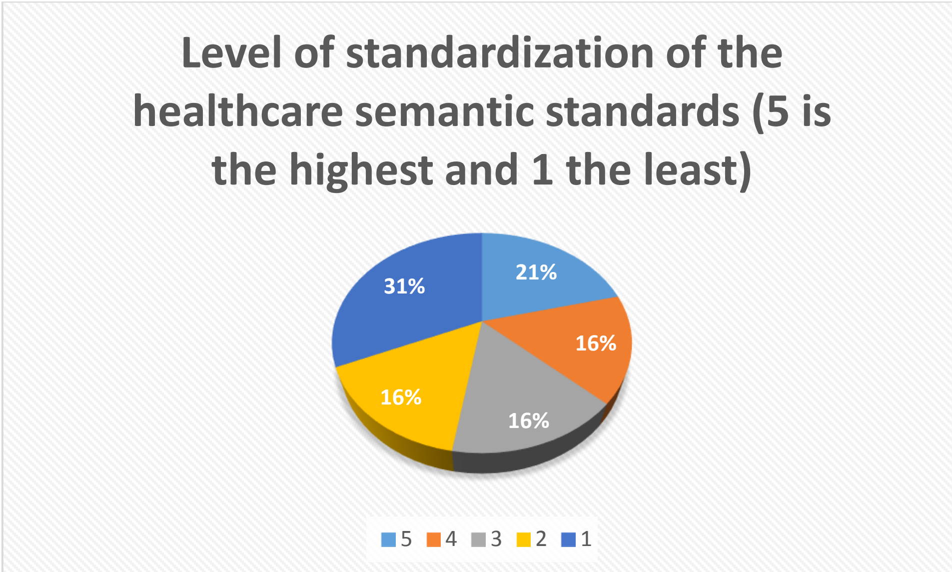

Figure 4.5: Level of Semantic standardization

Figure 4.5 above shows the level of standardization as regards semantic standards that allows for documentation in healthcare. 53% agree that there is a level of standardization in place that makes it easy to use.

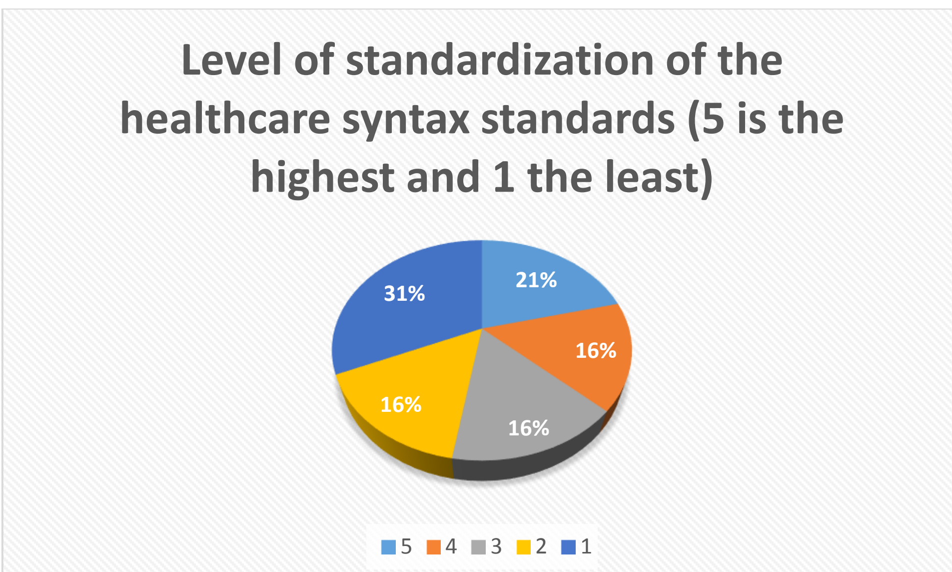

Figure 4.6: Level of Syntactic Standardization

Figure 4.6 above tries to understand the level of standardization as regards syntactic standards. 53% agree that there is a level of standardization in place that makes it easy to use. This is important in the implementation of systems during system development.

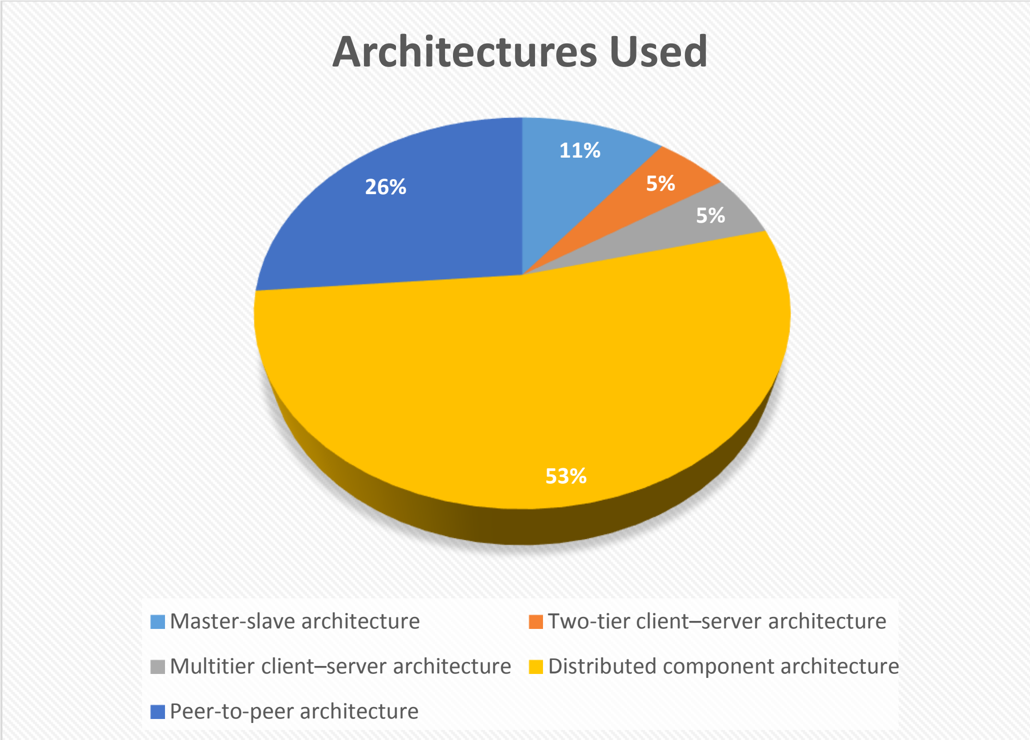

Figure 4.7: Architectures Used

Figure 4.7 above identifies the different architectures used in the development of a distributed healthcare system. Distributed component architecture is the most preferred at 53%.

4.3 Requirements Analysis

Requirements analysis provides a detailed description of services, features and constraints that the interoperability Healthcare API platform should address. This services, features and constraints can be fragmented into functional and non-functional requirements.

4.3.1 Functional Requirements

Functional requirements are functions, processes and capabilities that the API should be able to execute. These includes: login portal (Doctor logs in so as to initiate process), Patient Search (search patient from the referred healthcare institution), Review the data (a brief review of the data to determine diagnostics made and tests already carried out), manage patient illness (upon review, recommend tests / treatment and update patient record) and close patient record (logout).

4.3.2 Non-Functional Requirements

These are characteristics qualities that are not expressly needed but are implemented to make it interactive, user friendly and easy to use. These include security (allow secure log on such that only authorized personnel can log in and access patient's data), error reporting (aim at helping the users of the system manage errors they may face while trying to use the system e.g. wrong password), system availability (the system should be accessible if and when the doctor tries to access the API-based interoperability healthcare system), reliability (the system's ability to maintain a consistent state to its users), usability (ease of navigation and access to the systems' features), performance (ability of the system to multitask, support multiple access and requests) and scalability (ability to grow and expand with the needs of the users).

4.4 Proposed System Architecture

The actors of the developed API-based interoperable healthcare system platform include: user (this is anyone having access to patients' information with the aim of medicating them) and administrator (handle system management and administration) as seen in Figure 4.8.

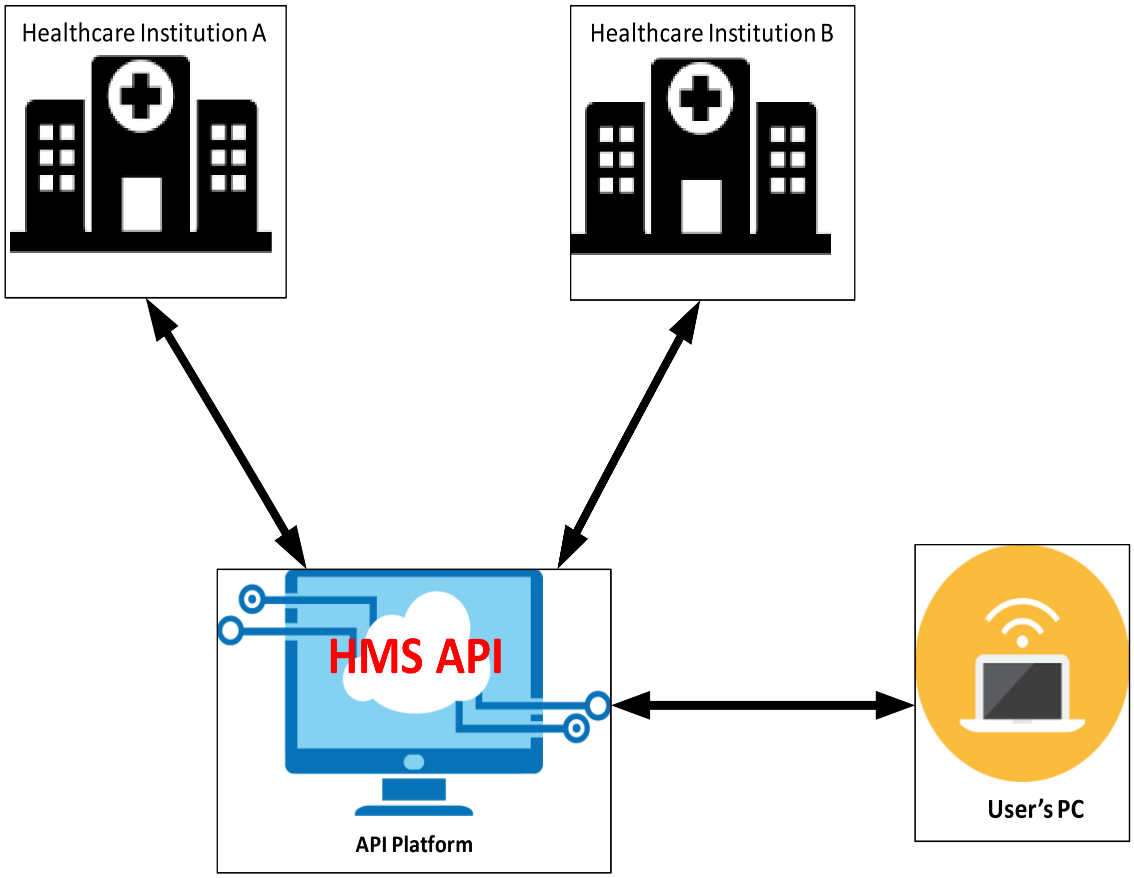

Figure 4.8: Proposed system architecture

4.4.1 Use Case Diagram

Figure 4.9 illustrates the main interactions that exist between the various subsystems and actors in the model.

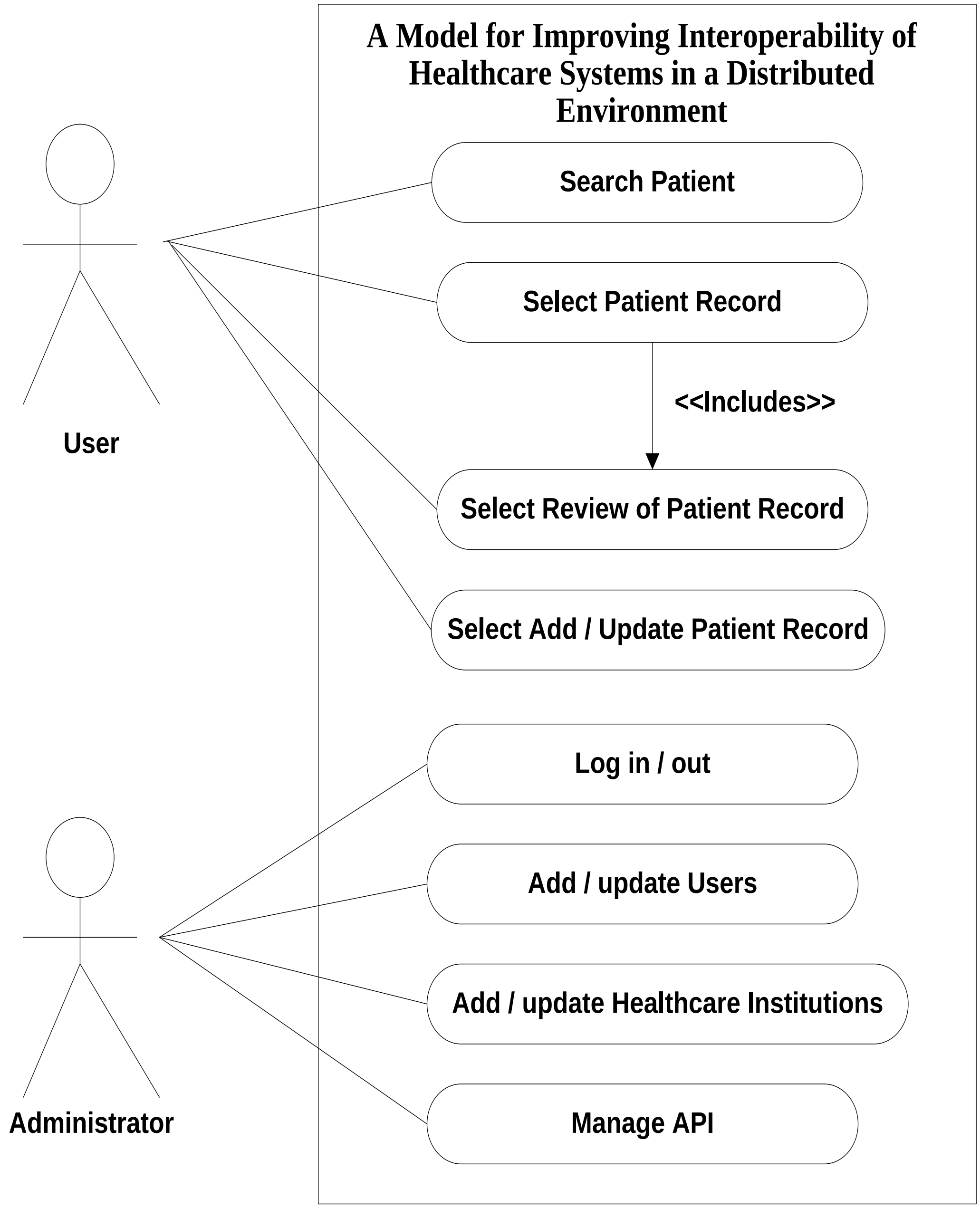

Figure 4.9: Use case diagram

The use cases are discussed below

Use Case: User Login

This use-case reports how Doctors access the system.

Preconditions:

Doctor is registered into the system and login credentials provided.

Post-conditions:

Doctor logs into the system with the credentials provided and access the API modules available to him.

Main Success Scenario:

i. Doctor is able to view the API modules available to him.

ii. Doctor selects doctors button to view list of available doctors.

iii. Doctor selects patients button to view list of patients in the main hospital system.

iv. Doctor selects Check-ups to be able to view patients' diagnostics i.e. internal and synchronized Check-ups.

v. Synchronization for referred patient file is via a telephone authorization code which when entered allows to view and update patient health record.

Alternative Flow: Authorization

At step 4, user selects to choose Check-ups button.

i. To access a referred patient's record, once phone must easily be accessible to allow for capturing authorization code.

ii. Doctor keys in the referred patient's authorization code that allows one to access patient's health record for review and further recommendations for treatment.

Use Case: Search Patient

This use case reports how one goes about finding a patient in the system

Precondition

Patient is registered either in the referral healthcare institution or in the referring one.

Post-conditions

Patient's file is found. If new, registration and diagnostic is done. If existing in the current hospital review of same illness else diagnose and update record. If referred patient, review file from referring hospital, diagnose and update record.

Main Success Scenario:

i. Doctor searches patient record (new, previous, referred) in the system.
ii. Doctor finds Patient.
iii. Doctor begins review, diagnostics and medication respectively.

Alternative Flow: Patient not Found

i. If search returns NILL, create patient a new in the system.

Use Case: Select Patient Record

This use case is tied to the search patient use case. A patient is selected to initiate the process of treatment.

Precondition:

Patient record exists.

Post condition:

Patient record file is opened for review and treatment.

Use Case: Select Review of Patient

This use case aims at reviewing a patient's health record to determine medical history.

Precondition:

Patient medical history is available under the patient's health record.

Post-condition:

Patient health record is reviewed or update as the doctor deems fit.

Use Case: Update patient record

This use case defines the updates to be carried out on a patient's record. The record can either be new or existing or referral.

Preconditions:

Patient medical history is available under the patient's health record.

Post-conditions:

Updated diagnostic and treatment.

4.4.2 Sequence Diagram

Key aspects of the system is that a patient referred from one healthcare institution to the next can have his record available after which review, diagnostic and treatment can start at the referral hospital. The diagram below highlights information flow from the time a patient's report to a doctor until one is reviewed, diagnosed and treated.

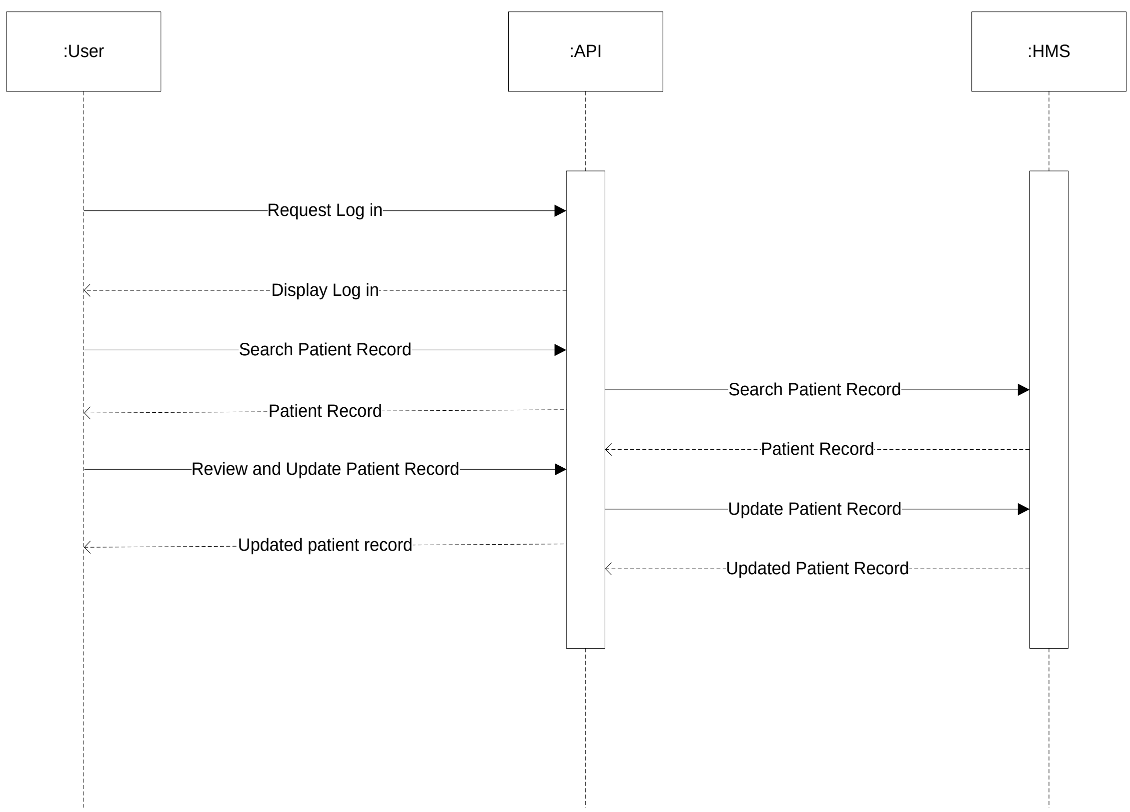

Figure 4.10: Sequence diagram

4.5 System Design

So as to be able to enhance the object requirements definition, Object-oriented design techniques were used. This information was captured at the point of system analysis and definition of objects that are design specific. The system design was based on user requirements and the study specifications.

4.5.1 Entity Relationship Diagram

The ERD is designed to capture the entities that are used in the database to capture, save and retrieve data to be accessed via a web interface that is power using an API. It seeks to show the specific tables and the resultant relationships.

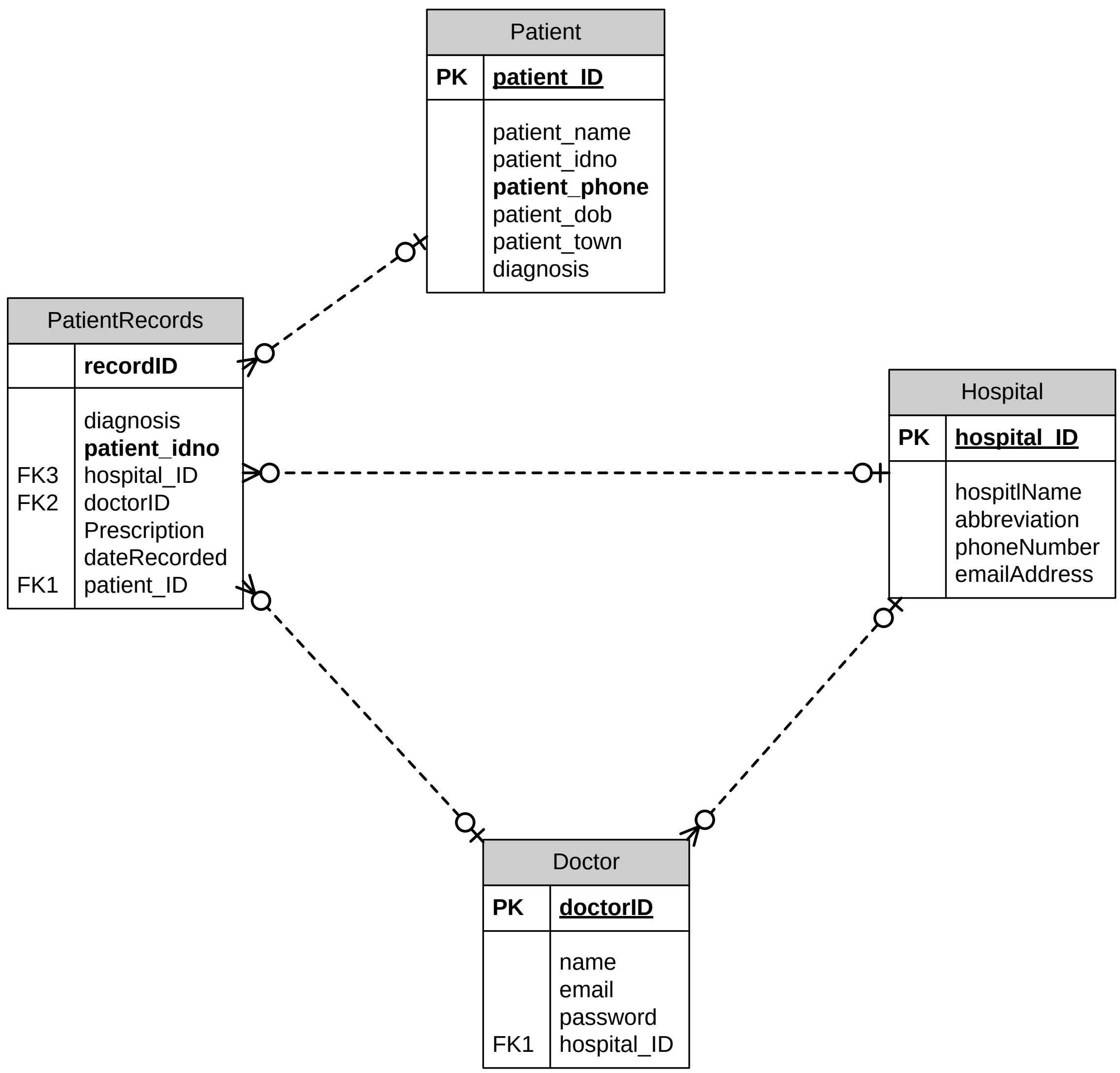

Figure 4.11: Entity Relationship Diagram

4.5.2 Design Class Diagram

This diagram aims showing the interactions between various classes in the system.

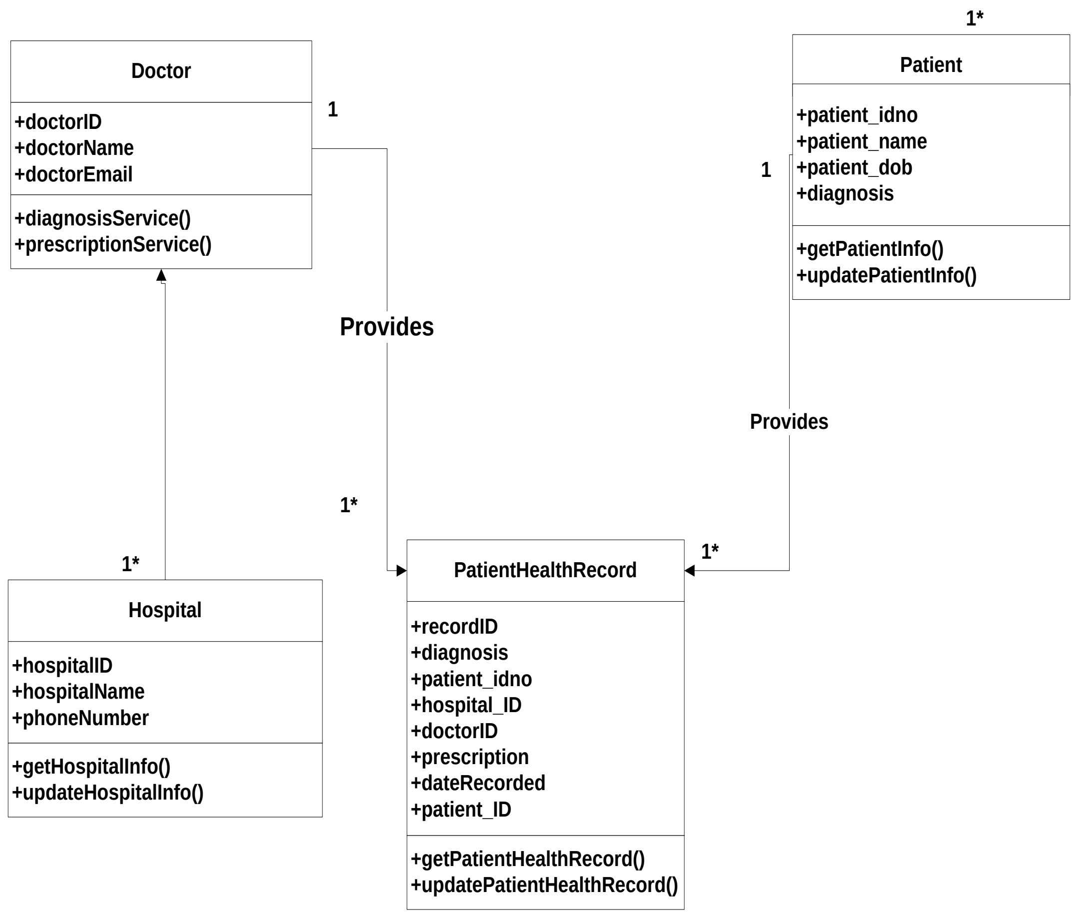

Figure 4.12: Design Class Diagram

4.5.3 Security Design

The security aspect considered by the design phase is that to access a patient's record, a token is sent to the patient's registered line which is used to access the patient's health record. This means that for anyone to access a patient's file, one needs to have the phone at hand and also determine beforehand, the hospitals from which the patient has been referred from. The other aspect of security is the hashing of the Doctors credentials and patient health records to curb against unauthorized access.

The Encryption Key is provided for as below:

This encryption key provided is used by the Illuminate encrypter service and is set to a random 32-character string. This is enforced before deployment of the application. The credentials are encrypted at the point of user creation.

Chapter 5 : System Implementation and Testing

5.1 Introduction

This chapter concentrates on the implementation and testing of the proposed API system. The focus on implementation is to bring a clear view of the different modules of the system and how each functions. Testing on the other hand looks at whether the system is usable and functional so as to determine if the set objectives have been achieved.

5.2 System Implementation

System implementation was done on a test server environment accessible by the healthcare practitioners for test purposes. Based on the literature review, the study sought to capture the specific user requirements. These requirements were identified and in section 4.3 they were defined as either functional or non-functional requirements. Once the requirements were determined, the design of the system was initiated at which point users were required to review the design and adopt it as having captured the actual requirements. Once the requirements and design were done, the development of the model was initiated with little input from the users. In the development phase of this system, care was taken to ensure that users are able to interact with a web based platform that rides on the developed API which makes different healthcare institutions interoperable without the user having to know how. To achieve this, a framework called Laravel and MySQL database were used. Having captured all the user requirements in the model, the system was ready for test and feedback.

The different modules of the system are explained below:

Figure 5.1 below provides a portal through which a patient can be registered into the system. This patient is registered if and only if one is new, else the patient record file is access to view medical history and run new diagnostics for treatment and medication purposes.

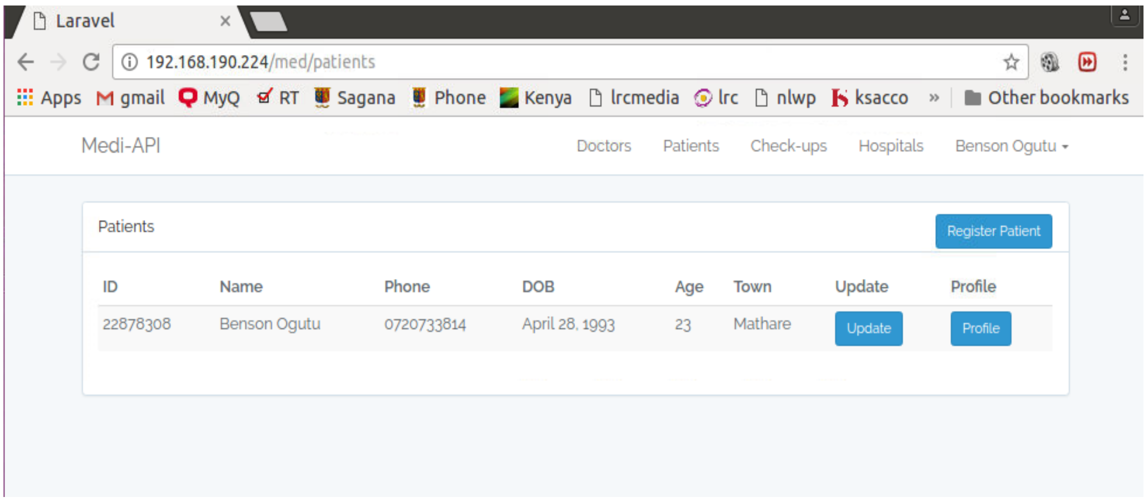

Figure 5.1: Patient registration

Figure 5.2 below captures the main objective of the study as it is the system's ability to capture internal checkups of a patient and synchronized checkups coming from referrals. Once the patient is seen by a referral doctor, the files are synchronized to capture data from both the referring and referral healthcare institutions.

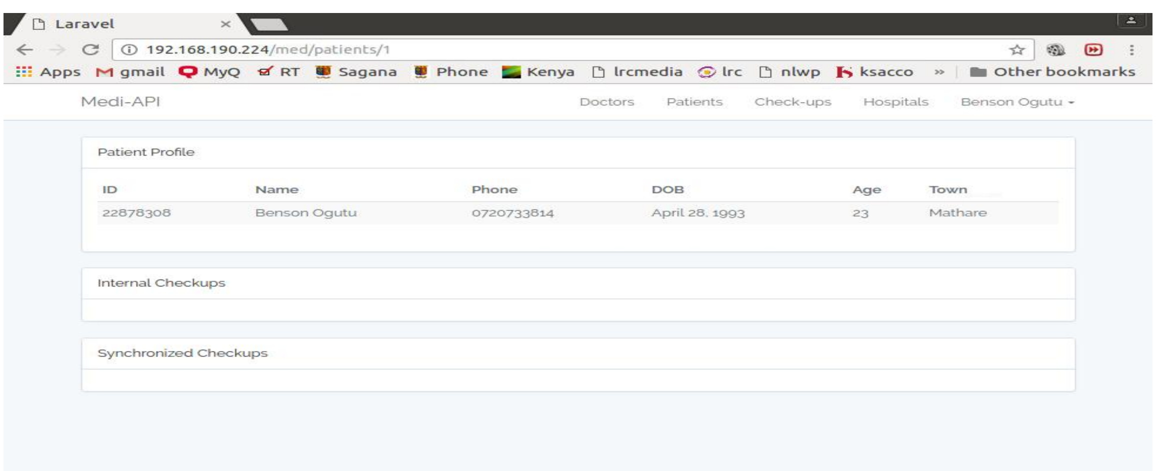

Figure 5.2: Internal and synchronized checkups

Figure 5.3 seeks to capture a doctor's documentation of the patient check-up record. This captures the doctor's diagnostic and prescription. These two are important fields since they are the important part of a patient's health record. If this data is captured well, it can make provision for efficient decision making.

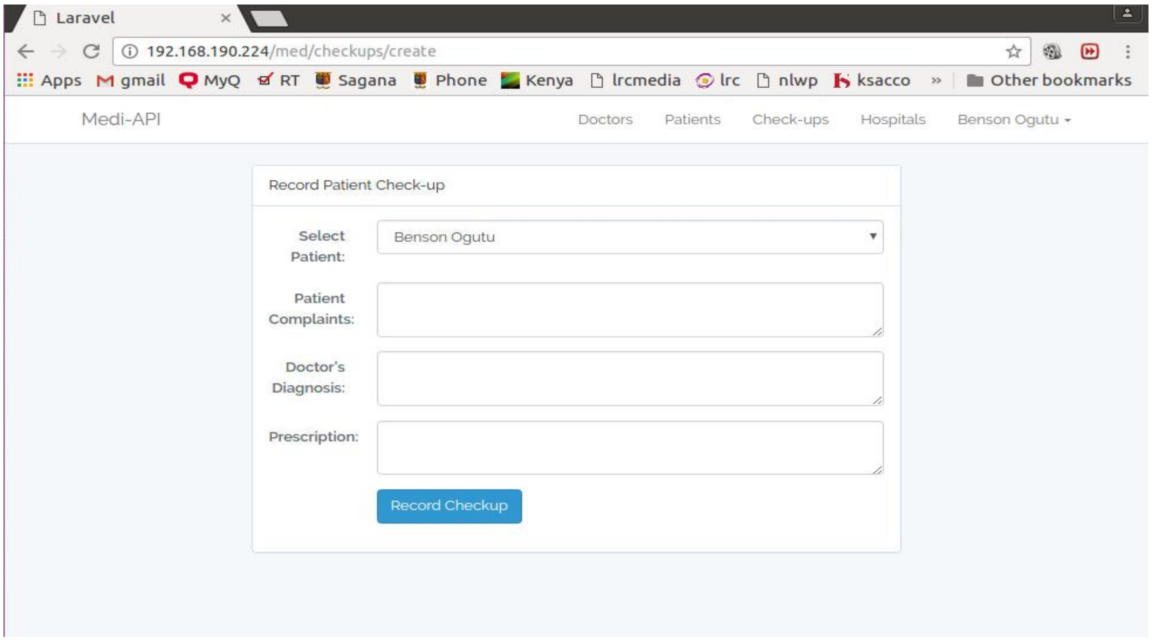

Figure 5.3: Patient checkup record

Creation of an API for a given hospital is as described in the code below:

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Hospital;
use Validator;
class HospitalsController extends Controller
{
    /**
     * Display a listing of the resource.
    */
    public function index()
    {
        $hospitals = Hospital::all();
        return view('hospitals',compact ('hospitals'));
    }
    public function create ()
    {
        return view('create-hospital');
    }
    /**
     * Store a newly created resource in storage.
   */
```

```
public function store (Request $request)
{
        $validator = Validator::make($request->all(), [
        'name' => 'required|unique:hospitals',
        'abbreviation' => 'required|unique:hospitals',
        'api' => 'required|unique:hospitals'
        ]);
    if ($validator->fails ()) {
        return  back ()
        ->withErrors ($validator)
        ->withInput ();
    }
    Hospital::create(['name'=>$request->get('name'),'abbreviation'=>$request-
>get('abbreviation'),'api'=>$request->get('api')]);
    return redirect('hospitals');
}
}
```

The code above allows the creation of an instance of another hospital by the API via the *class hospitalController*. A hospital system is linked to the API by capturing the required fields and then registered as part of the existing systems in the parent healthcare management system. This allows for ease of access of the referring hospital by the doctor at the point of patient review. This line of code allows one to be able to capture the name of the hospital, abbreviation and the link to that specific hospital. The link allows one to mimic logging into the main system.

API Route:

Once the hospital is registered in the system, an API route is created for it as seen below. API routes are register here for your application. These routes are loaded by the RouteServiceProvider within a group which is assigned the "api" middleware group:

```
<?php
use Illuminate\Http\Request;
/* API Routes */
Route::middleware('auth:api')->get('/user', function (Request $request) {
   return $request->user();
});
```

The routes for the application are defined as below:

```
public function map()
   {
      $this->mapApiRoutes(); //API maps the client to the host
      $this->mapWebRoutes(); maps the different links to be accessed by the API
      //
   }
```

The web routes for the applications are:

```
protected function mapWebRoutes()
   {
      Route::middleware('web') // the middleware / API is design to act as a mirror
between the two or more systems that seek to exchange data seamlessly.
         ->namespace($this->namespace)
         ->group(base_path('routes/web.php'));
```

```
}
```

The API routes for the applications are:

```
protected function mapApiRoutes()
{
    Route::prefix('api')
        ->middleware('api') // facilitates the maping of the different healthcare institutions at the point of the accessing the system.
        ->namespace($this->namespace)
        ->group(base_path('routes/api.php'));
}
```

Security:

This encryption key provided is used by the Illuminate encrypter service and is set to a random 32-character string. This is enforced before deployment of the application.

```
“'key' => env('APP_KEY'),
 'cipher' => 'AES-256-CBC',”
```

At the point of user creation, the credentials are encrypted as shown below:

```
protected function create (array $data)
 {
    return User :: create([
       'name' => $data['name'],
       'email' => $data['email'],
       'password' => bcrypt($data['password']),
    ]);
```

Algorithm to access the patient's health record is as below:

Session starts;

is doctor logged in? No

ask to log in, else;

search for patient using phone number, if present,

token is sent to patient's phone for validation process else register patient

if token is valid, access patient health record for review and update

end session.

Figure 5.4 allows the linking of the system of the healthcare institution that one seeks to implement interoperability. This allows for the API to look for the specific fields thereby allowing data to be easily accessible across systems.

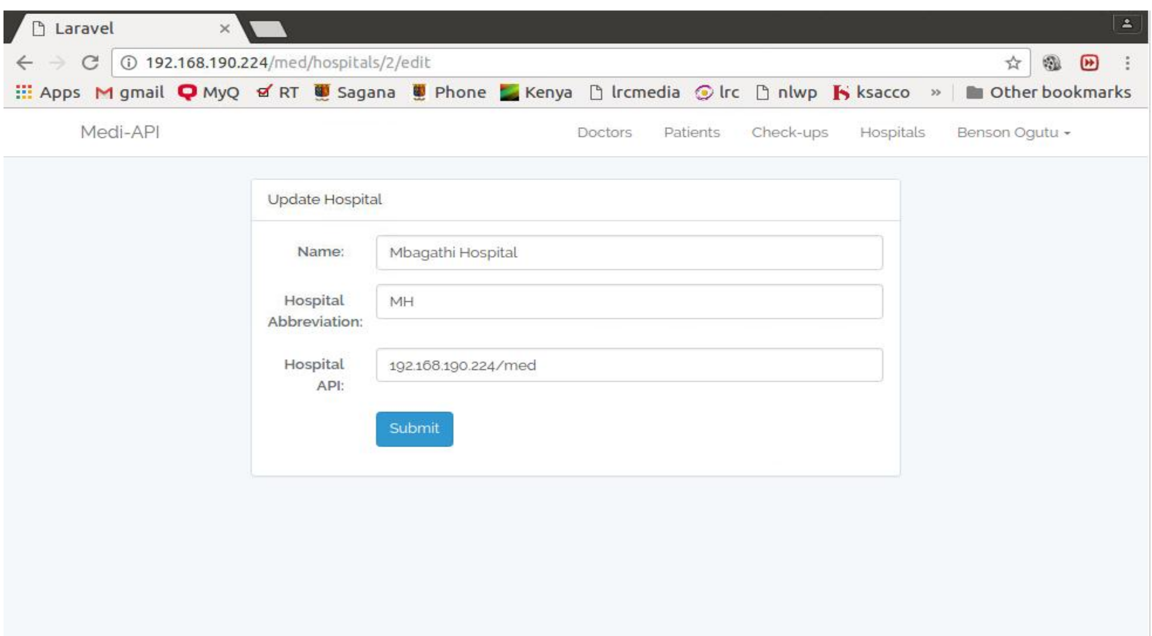

Figure 5.4: Linking hospitals

Once a hospital has been linked to the main hospital, one has access to the system and its patients as shown in Figure 5.5 below:

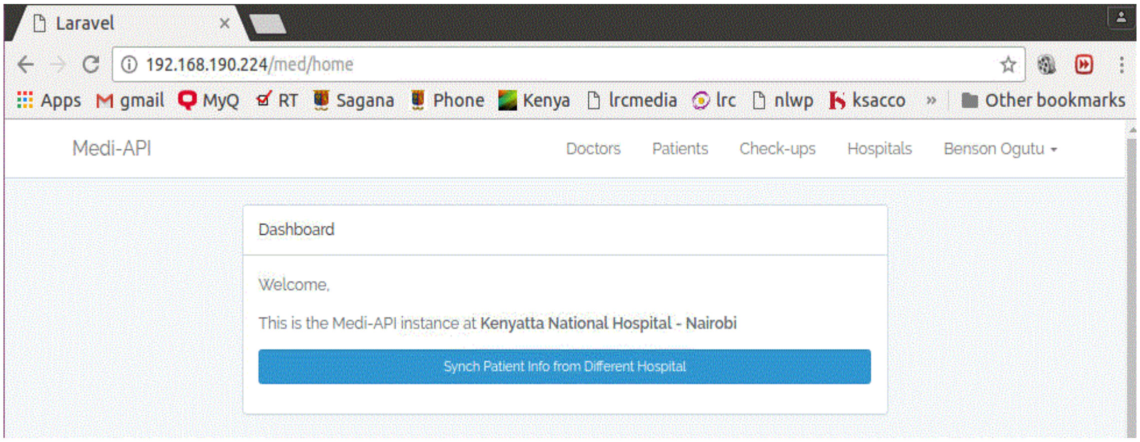

Figure 5.5: Synch patient info from different hospital

By clicking on the link: ***synch patient info from different hospital***, shown in Figure 5.5 above, one automatically initiates the process of looking up for the patient by first selecting the patient using one's phone number and selecting the hospital from which ones seeks to access the data as shown in Figure 5.6 below.

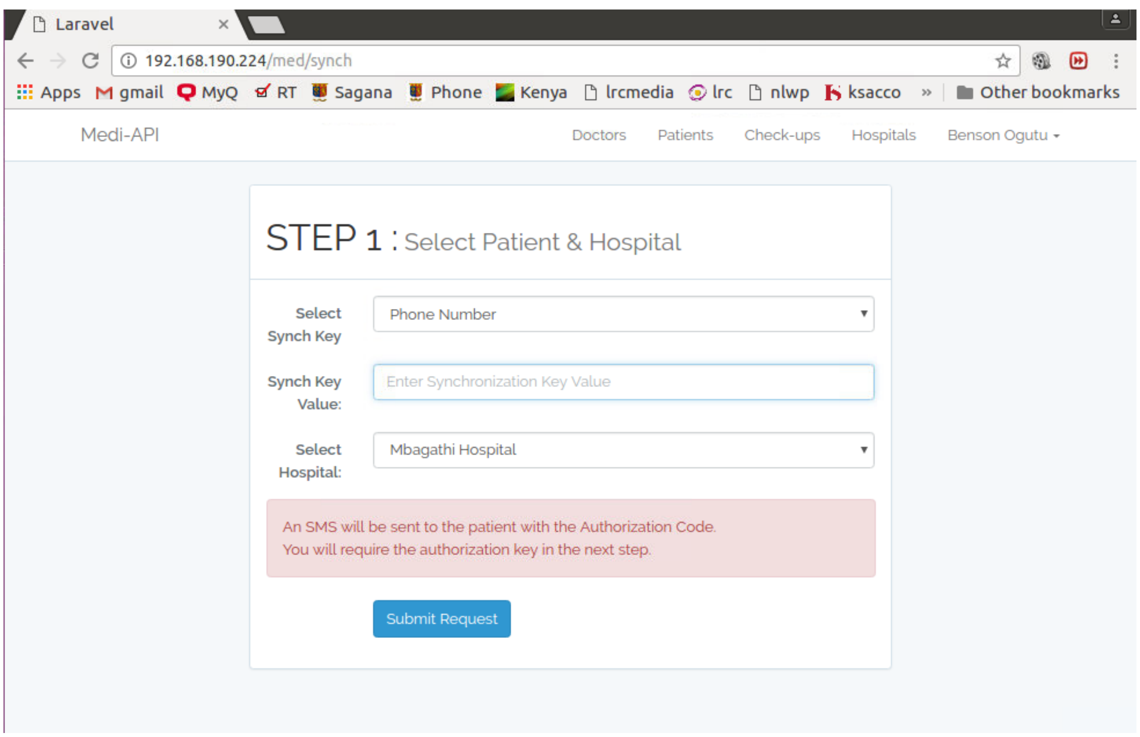

Figure 5.6: Accessing referral patient health record

Figure 5.6 above shows how a patient record from one hospital is accessed from the referral hospital. The patient's phone number has to be furnished for it is through it that the authorization code will be sent. Also, the system allows for the selection of the referral hospital that a patient is from for purposes of treatment and medication.

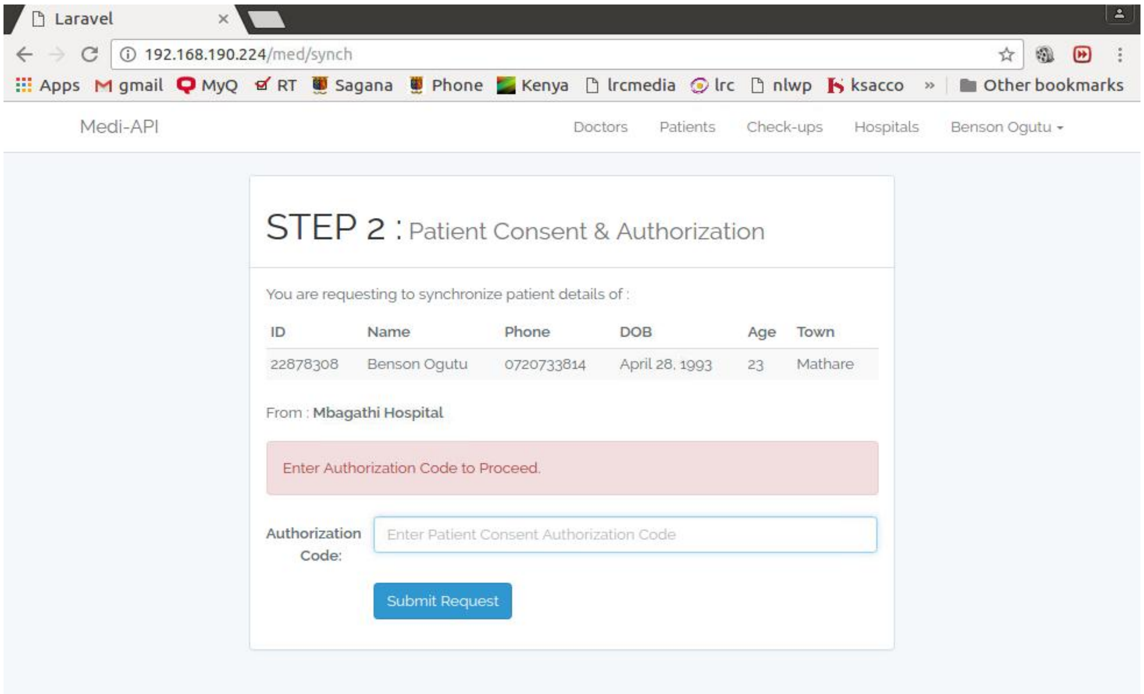

Figure 5.7: Validation process via patient's phone

Figure 5.7 above shows the validation process by which a patient's health record cannot be access except that one has access to the patient's telephone and number. Once validated access to the synchronized information is easily accessible as see in Figure 5.8 below:

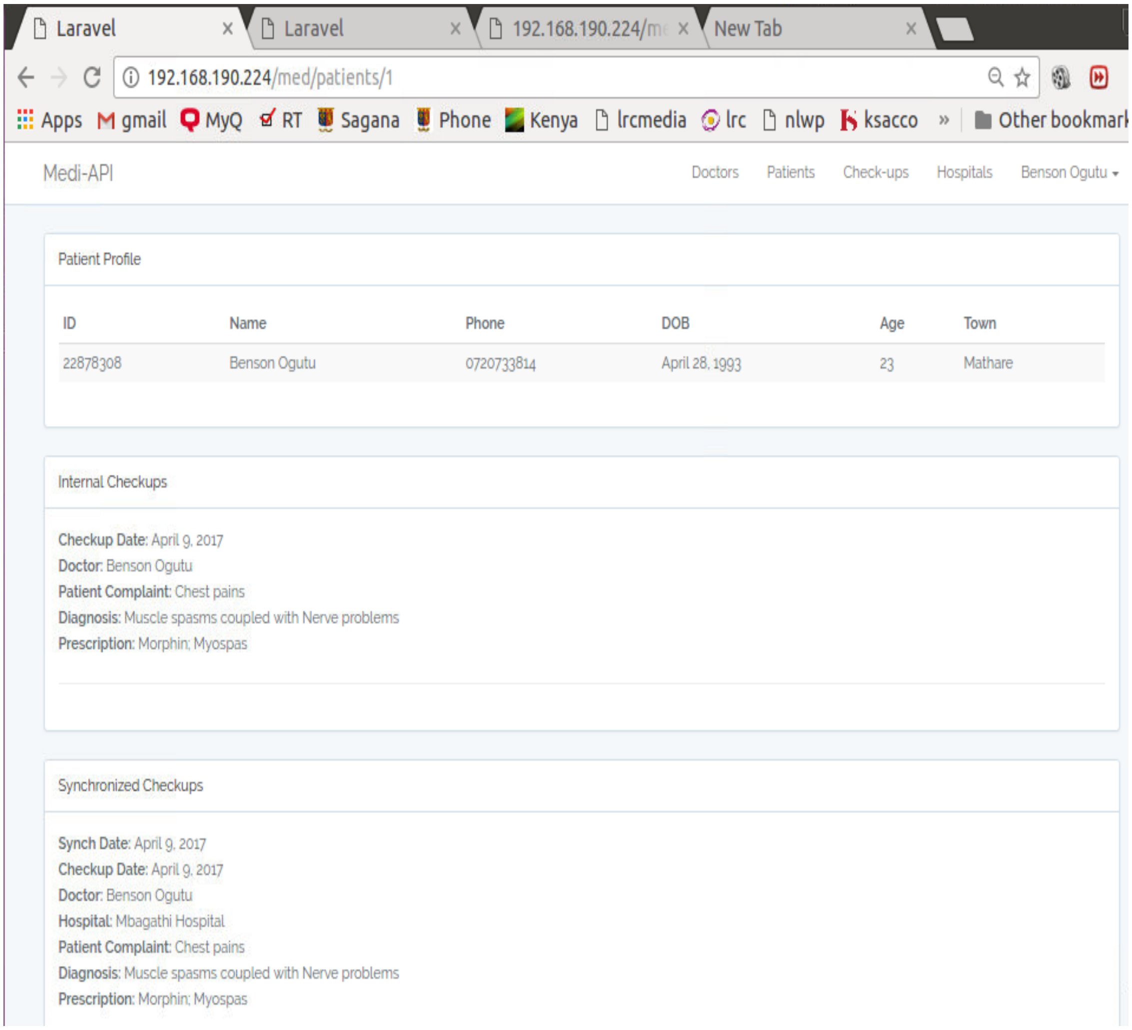

Figure 5.8: Synchronized data

The system has been configured to ensure that it is highly responsive in alerting any errors that are user based. This alerts allow the user to determine what it is that is hindering one from fully accessing and utilizing the functionality of the system. As such, Figure 5.9 below demonstrates the responsiveness of the developed system.

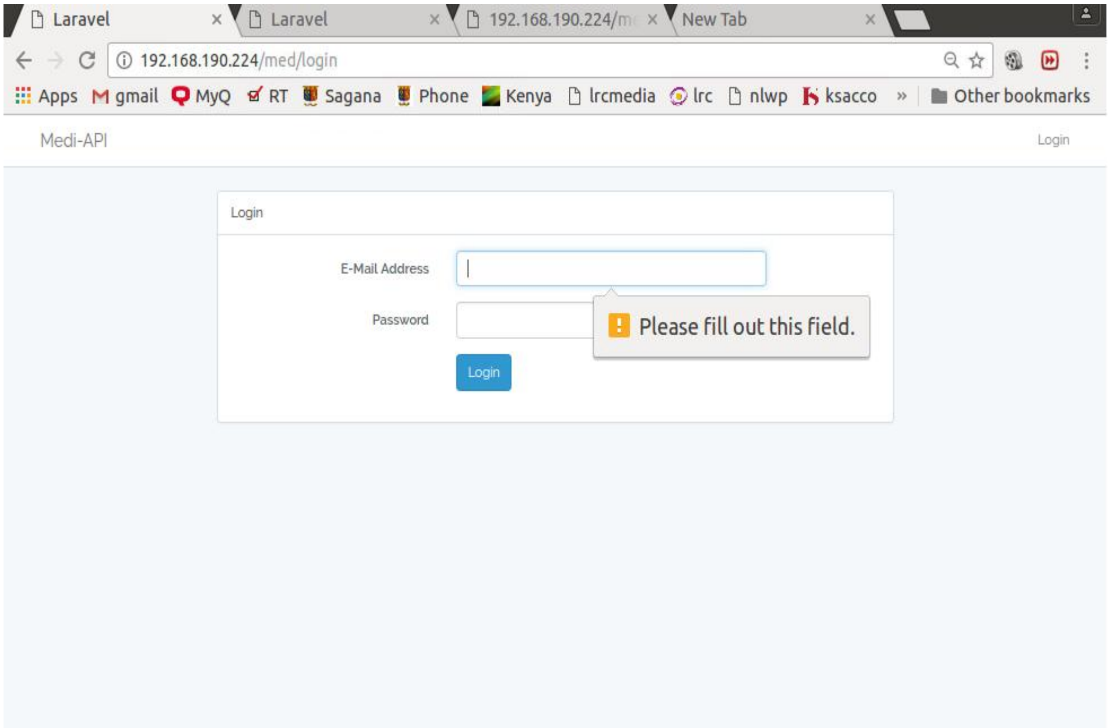

Figure 5.9: System response

API Code Snippet:

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Patient;

use App\Hospital;

use App\Checkup;

use App\SychdData;

use Log;

use Validator;

class APIController extends Controller
```

```
{
   public function index()
   {
          $hospitals = Hospital::where('id','<>',1)->get();
          return view("sych-select-patient",compact('patients','hospitals'));
   }
public function dosynchdata($authkey,$patientkey,$patientvalue){
   //check if authkey is okay.
   $patient = Patient::where($patientkey,$patientvalue)->first();
   $checkups = Checkup::join('patients','patient_id','=','patient_key')
      ->join('users','id','=','doctor_key')->where('patient_key',$patient->patient_id)
      ->select('users.name as doctor_name','users.email as doctor_email','checkup_complaint','checkup_diagnosis','checkup_prescription','checkup_date')
      ->get();
    return array("patient_info"=>$patient,"patient_checkup_info"=>$checkups);
}
   public function synchdata(Request $request)
   {
          $validator = Validator::make($request->all(), [
          'patient_key' => 'required',
          'patient_authcode' => 'required',      //token provided for via patient's phone
```

```
    'hospital_key' => 'required',
    'synch_key' => 'required'
    ]);
  if ($validator->fails()) {
    return  back()
    ->withErrors($validator)
    ->withInput();
  }
  $authcode = $request->get('patient_authcode');        //supply token gotten via patient's phone.
  $hospital = Hospital::find($request->get('hospital_key'));
  $hospital_ws = $hospital->api;
  $patient_key = $request->get('patient_key');
  $synch_key = strtoupper($request->get('synch_key'));
  $raw_synchdata = file_get_contents($hospital_ws."/".$authcode."/".$synch_key."/".$patient_key);
  $synchdata = json_decode($raw_synchdata);
  //Add User
  $synched_patient = $synchdata->patient_info;
  $pn = Patient::where($synch_key,$patient_key)->first();
  if(null == $pn){
    $patient = Patient::create([
```

```
        'patient_name'=>$synched_patient->patient_name,
        'patient_idno'=>$synched_patient->patient_idno,
        'patient_phone'=>$synched_patient->patient_phone,
        'patient_dob'=>$synched_patient->patient_dob,
        'patient_town'=>$synched_patient->patient_town
    ]);
}else{
    $patient =$pn;
}
//Save sysnchronized
foreach( $synchdata->patient_checkup_info as $synched_checkups){
     try{
    SychdData::create(
        [
            'synch_patient_id'=>$patient->patient_id,
            'synch_doctor'=>$synched_checkups->doctor_name,
            'synch_doctor_email'=>$synched_checkups->doctor_email,
            'synch_hospital'=>$hospital->name,
            'synch_complaint'=>$synched_checkups->checkup_complaint,
            'synch_diagnosis'=>$synched_checkups->checkup_diagnosis,
            'synch_prescription'=>$synched_checkups->checkup_prescription,
```

```
            'synch_checkup_date'=>$synched_checkups->checkup_date
          ]
          );
    }catch(Exception $e){
      Log::info("synchdata already exits");
    }
    }
    return redirect("patients/".$patient->patient_id);
}
public function getSelectedPatient(Request $request)
{
      $validator = Validator::make($request->all(), [
      'synch_key' => 'required',
      'synch_key_value' => 'required',
      'hospital_key' => 'required'
      ]);
    if ($validator->fails()) {
      return  back()
      ->withErrors($validator)
      ->withInput();
    }
```

```
$val = $request->get('synch_key_value');

$thekey = $request->get('synch_key');

$key_used =null;

switch ($thekey) {

    case 'patient_idno':

        $key_used = "Identity Card Number";

        break;

    case 'patient_phone':

        $key_used = "Phone Number";

        break;

}

    $patient = Patient::where($thekey,$val)->first();

    $hospital = Hospital::find($request->get('hospital_key'));

    if(null != $patient && null != $hospital){

        return view('synch-consent',compact('patient','hospital','thekey','val'));

    }else{

        return redirect()->back()->withInput()->withErrors(['Not Found' => 'No Patient exists with '.$key_used.'  : '.$val.'. Check your number and try again']);

    }

}

}
```

5.3 System Testing

As discussed in Chapter Three, Agile methodology was applied in this study and as such, agile testing was deployed during the testing of the software to check both on the bugs and system performance issues. The beauty of doing system testing before full deployment is so that any errors that had not been foreseen to be removed and live the system in a more improved and usable fashion. The constant back and forth with the users make the system more robust with little or no error (Hendrickson, 2008).

5.4. Usability Testing Results

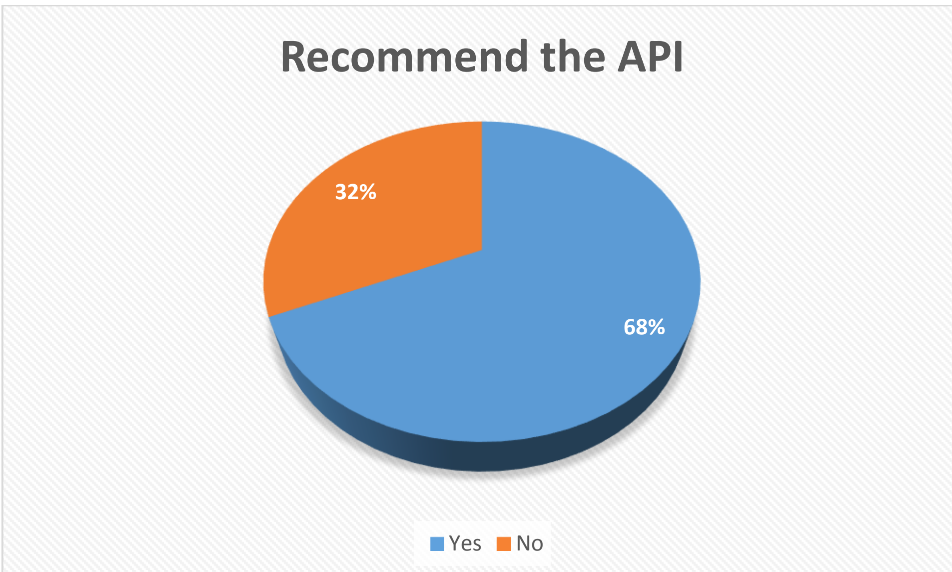

Figure 5.10: Recommend the API

From Figure 5.10, 68% of the respondents said that they would easily recommend the API to be used across hospitals so as to help fasttrack process and improve decision making.

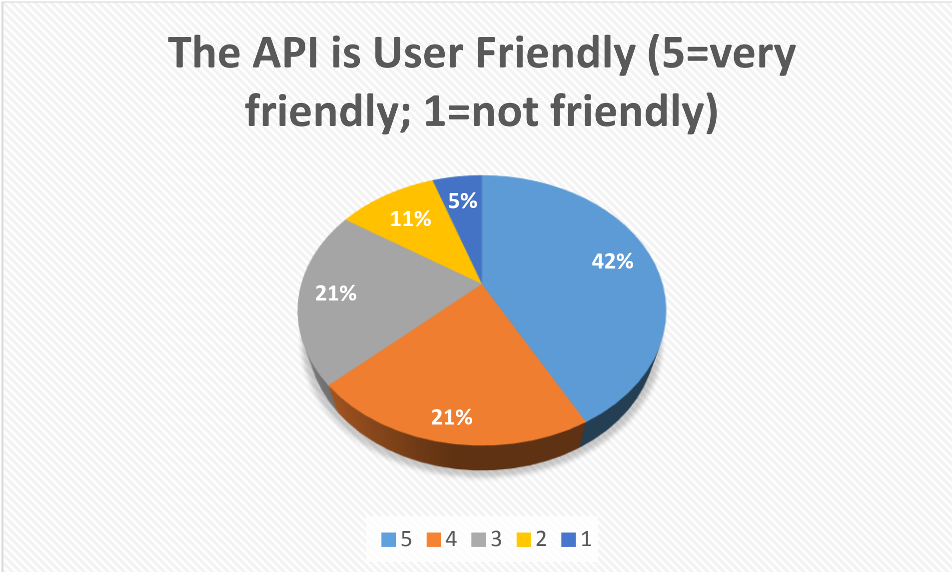

Figure 5.11: User friendly

From Figure 5.11, 84% of the respondents agree that the API is user friendly and would be easy to learn and adopt as one of their key processes.

Chapter 6 : Discussion

6.1 Introduction

The findings obtained during the research formed the basis upon which an API platform for improving interoperability of healthcare systems in a distributed environment was developed. The API was further tested to establish that it functions according to the research. The basis of the analysis done in this chapter is the findings with regard to the stated research objectives and the extent to which the findings are in agreement with the literature review.

6.2 Factors affecting interoperability in healthcare systems

From the data collected and presented in section 4.3, key to this factors include standardization issues such as syntax and semantics, system architectures, usage, portability of data, and CIA of data. This objective is well covered in Chapter Two of this study.

6.3 Interoperability Healthcare Standards and Architectures for a Distributed Environment

This being the second objectives, it capitalized on the strength gained in chapter Two under literature review. Two standards stand out when it comes to matters healthcare standardization and these are semantic and syntax standards. Semantics is aimed at the documentation of the system whereas syntax aims at managing the actual system development process. The architectures discussed in this study include master-slave, two-tier client-server, multi-tier client server, distributed components, peer-to-peer and service oriented architectures. This standards and architectures were important to this study in the sense that they allowed for ease of design and implementation of the API.

6.4 API Model for Enhanced Interoperability

An API basically aims at allowing two disparate systems to communicate. Objective three of the study sought to have the API designed and developed with respect to all the combined literature review which was quite elaborate. Based on the research findings, users are more reliant on a web based API that is accessible from anywhere in the organization.

6.5 API Model for Enhanced Interoperability Application Testing

The aim of this objective was to determine the usability of the system by the users and also check on how well it meets the necessary functional and nonfunctional. All bugs and fixes were identified and communicated.

6.6 Advantages of the API as Compared to the Current System

The API-Based model for Interoperability in distributed healthcare environment allows for an electronic patient file. This far outweighs the current system where any referrals are done by word of mount and there is no treatment sheet that can be reference to. Lack of referencing during a referral increase costs of seeking medical attention and at the same time it delays decision making process.

6.7 Disadvantages of the API Application

The successful implementation of this API is that internet has to be available else it becomes impossible to implement it. Both systems to be integrated for interoperability have to be online which rather inconvenience the organizations having to invest on infrastructure they had not alluded to that.

Chapter 7 : Conclusions, Recommendations and Future Work

7.1 Introduction

This chapter provides the conclusion of the study based on the set objectives, recommendations as derived throughout the course of this research and future work as far as it relates to this study area.

7.1 Conclusion

From the study, ease of access, data portability, data confidentiality, security and integrity, data formats, file sharing, cost reduction and scalability of systems were the factors identified to be critical in the establishment of an interoperable healthcare system in a distributed healthcare environment. Ease of access was rated highly at 47% followed by data confidentiality, integrity and security at 16%. The standards identified from the study were summarized as semantic standards syntactic standards. The level of standardization of both was at 53% for each which shows that there is a level of structuring taking place that is key to the implementation of interoperability in healthcare. The architectures identified by the study included master-slave architecture, two-tier client-server architecture, multi-tier client-server architecture, distributed component architecture and peer-to-peer architecture. Distributed component architecture was the preferred architecture at 53%. The API model developed through the study employed the use of agile methodology where fast iterations are done with frequent user feedback allowing for user participation and faster development.

7.2 Recommendations

Interoperability in healthcare is vital as it allows for faster processes on the part of the healthcare institutions while helping reduce costs on the part of the patients. With the automation levels seen in most healthcare institutions, the study recommends the full standardization of both syntax and semantic standards. This is important as it allows for ease of interoperability. Also, standards need be set for the implementation of interoperability healthcare systems.

7.3 Future Work

Based on the findings of this study, the study forecast further development and input in this field to include cloud-based interoperability solution that allows for seamless operations and can ensure that all healthcare systems are centrally managed and information be easily accessible across all healthcare institutions for the specific system users.

References

Adams, D., Nelson, R., & Todd, P. (1992). "*Perceived usefulness, ease of use, and usage of information technology*: *A replication*", MIS Quarterly 16: 227–247, doi:10.2307/249577.

Ajzen, I., & Fishbein, M. (1980). *Understanding attitudes and predicting social behavior*. Englewood Cliffs, NJ: Prentice-Hall.

Alabama State University. (2013). *The Conceptual Framework. College of Education. Alabama State University; Corvellec, Hervé, ed. What is Theory? Answers from the Social and Cultural Sciences*. Stockholm: Copenhagen Business School Press.

Ryan, A. (2006). "*Towards Semantic Interoperability in Healthcare: Ontology Mapping from SNOMED-CT to HL7 version 3*"., Conferences in Research and Practice in Information Technology (CRPIT)," vol. 72, pp.1-6.

ASTM International. (2012). Retrieved from ASTM: https://www.astm.org/Standards/E1384.htm

Bacon, J., & Moody, K. (2002). *Toward open secure widely distributed services*. Magazine Communications of the ACM, Vol. 45 pp.6.

Bagozzi, R. (2007). "*The legacy of the technology acceptance model and a proposal for a paradigm shift*.", Journal of the Association for Information Systems 8 (4): 244–254.

Bickman, L. (1989). "*Barriers to the use of program theory*." Evaluation and program planning. 12 (4), 387-390. Pergamon Press plc.

Bickman, L & Rog, D.J. (2008). *Handbook of Applied Social Research Methods*. SAGE.

International Organization for Standardization. (1995). *Different forms of transparency in a distributed system*. Retrieved from http://csis.pace.edu/~marchese/CS865/Lectures/Chap1/Chapter1a.htm.

California State University. (2012). *PPA 696 Research Methods Data Collection Strategies II: Qualitative Research*. Retrieved from Qualitative Research: https://web.csulb.edu/~msaintg/ppa696/696quali.htm.

Councill, W. T. & Heineman, G. T. (2001). *Definition of a Software Component and its Elements'. In Component-based Software Engineering*. (ed.). Boston: Addison-Wesley.

CPrime. (2014). *Hybrid Projects: How Can Waterfall and Agile Work Together*. Retrieved from CPrime https://www.cprime.com/2014/05/hybrid-projects-how-can-waterfall-and-agile-work-together/

Dana P. & Ciprian C. (2014). *Towards a Cross Platform Cloud API, Components for Cloud Federation*. Massimiliano Rak. Second University of Naples, Italy.

Davis, F. D. (1989), "*Perceived usefulness, perceived ease of use, and user acceptance of information technology*", MIS Quarterly 13 (3): 319–340, doi:10.2307/249008.

Erl, T. (2005). *Service-Oriented Architecture: Concepts, Technology and Design*. Upper Saddle River, NJ: Prentice Hall.

Erl, T. (2004). *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*. Upper Saddle River, NJ: Prentice Hall.

Newcomer, E. & Lomow, G. (2005). *Understanding SOA with Web Services*. Boston, Addison-Wesley.

ESHI. (2000). *CEN/TC 251 European Standardization of Health Informatics*. Retrieved from https://www.hl7.org/documentcenter/public_temp_4C7893FF-1C23-BA17-0CA260E5284EE34D/wg/secure/View_01.htm/.

Geert, J. (2017) *API Components*. Thoughts on RESTful API design. Release 1.0. Retrieved from https://www.redhat.com/archives/rest-practices/2011-August/pdfa1nfEjPMmT.pdf

Golafshani, N. (2003). *Understanding Reliability and Validity in Qualitative Research*. The Qualitative Report, 8(4), 597-606. Retrieved from http://www.nova.edu/ssss/QR/QR8-4/golafshani.pdf

Hahn, C., Jacobi, S., & Raber, D. (2010). *Enhancing the interoperability between multiagent systems and service-oriented architectures through a model-driven approach, web intelligence and intelligent agent technology (WI-IAT).* IEEE/WIC/ACM International Conference, vol. 2, p. 415.

Healthcare Information and Management Systems Society. (2006). Retrieved from http://www.himss.org/library/compatibility-of-data-exchange/.

Healthcare Information and Management Systems Society. (2013). *Learn what makes up the Health Interoperability Ecosystem*. Retrieved from http://www.himss.org/library/interoperability-standards/what-is-interoperability.

International Organization for Standardization. 2012. Retrieved from http://www.iso.org.

Iroju, O., Soriyan, A., Gambo, I., & Olaleke, J. (2013). *Interoperability in Healthcare: Benefits, Challenges and Resolutions*. International Journal of Innovative and Applied Studies, Morocco, Vol. 3pp 262-270.

Kombo, D. K., & Tromp, D. L. (2006). *Proposal and thesis writing: An introduction*. Nairobi: Paulines Publications Africa.

Kothari, C. (2004). *Research Methodology: Methods and Techniques*. 2nd Rev. ed. New Age International.

Maciel, R.S.P., & David, J.M.N., (2007). WGWSOA: *a service-oriented middleware architecture to support groupware interoperability*. IEEE 11th International Conference in Computer Supported Cooperative Work in Design (CSCWD), p. 556.

March, J. & Olsen J. (1984). *The New Institutionalism: Organizational Factors in Political Life*. American Political Science Review 78, 738–49.

March, J. & Olsen J. (1989). *Rediscovering Institutions*. New York: Free Press.

March, J. & Olsen J. (1996). *Democratic Governance*. New York: Free Press.

McDougall, P. (2000). '*The Power of Peer-To-Peer'*. Information Week (August 28th, 2000).

Ministry of Medical Services and Ministry of Public Health and Sanitation. (2010). *Standards and Guidelines for Electronic Medical Record Systems in Kenya*.

Mugenda, O.M & Mugenda, A. G. (2003). *Research Methods, Quantitative and Qualitative Approaches, Revised edition*. Nairobi: Act Press.

National Electrical Manufacturers Association (NEMA), 2012. http://www.nema.org/Standards/Pages/All-Standards.aspx

Neuman, B. C. (1994). '*Scale in Distributed Systems*'. In Readings in Distributed Computing Systems. Casavant, T. and Singal, M. (ed.). Los Alamitos, Calif.: IEEE Computer Society Press.

Oram, A. (2001). '*Peer-to-Peer: Harnessing the Benefits of a Disruptive Technology*'. ISBN: 0-596-00110-X, 448 pages.

Orfali, R., Harkey, D. & Edwards, J. (1997). *Instant CORBA*. Chichester, UK: John Wiley & Sons.

Pedersen, S., & Hasselbring, W. (2004). *Interoperabilität für informationssysteme im gesundheitswesen auf basis medizinischer standards*. Informatik – Forschung und Entwicklung, 18(3–4), 174–188.

Peters, B. (2000). *Institutional Theory: Problems and Prospects*, ISSN: 1605-8003.

PHP. (2017). *PHP: Hypertext Preprocessor*. Retrieved from http://php.net/manual/en/index.php

Roy, T.F. (2000). *Representational State Transfer (REST)*. Retrieved from http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Schweiger, A., Sunyaev, A., Leimeister, J., & Krcmar, H. (2007). *"Information Systems and Healthcare XX: Toward Seamless Healthcare with Software Agents,"* Communications of the Association for Information Systems: Vol. 19, Article 33.

Shaker H. El-Sappagh & Samir El-Masri. (2013). *A distributed clinical decision support system architecture*. Journal of King Saud University - Computer and Information Sciences, Volume 26, Issue 1, January 2014, Pages 69–78.

Steinmo, S., Thelen K. & Longstreth F. (1992) *Structuring Politics: Historical Institutionalism in Comparative Analysis*. Cambridge: Cambridge University Press.

Sunyaev, A., Leimeister, J., Schweiger, A., & Krcmar, H. (2008): *IT-Standards and Standardization Approaches in Healthcare*. In: Encyclopedia of Healthcare Information Systems. Idea Group, Erscheinungsjahr.

Tannenbaum A., & Maarten V. (2007). *Distributed Systems: Principles and Paradigm*. Pearson Prentice Hall.

The American Medical Association. (2012). Retrieved from http://www.ama-assn.org

The European Committee for Standardization. (2012). *E-health Standardization*. Retrieved from https://www.cencenelec.eu/standards/Sectors/healthcare/eHealth/Pages/default.aspx

The International Health Terminology Standards Development Organization (IHTSDO). (2012). Health Terminology Standards. Retrieved from www.ihtsdo.org./health-terminology-standards.

The Health Level Seven. (2012). *HL7 Standards*. Retrieved from http://www.hl7.org/hl7-standards.

The Regenstrief Institute, Inc. (2012). Cost reduction in healthcare. Retrieved from http://www.regenstrief.org/cost-reduction-in-healthcare.

The World Health Organization. (2012). *Enabling Patients using ICT*. Retrieved from http://www.who.int

US Department of Health & Human Services. Centers for Medicare & Medicaid Services. (2012). Retrieved from www.cms.gov

US Department of Health & Human Services, The office of the National Coordinator for Health Information Technology. (2012). Retrieved from http://www.hhs.gov/healthit

Vroom, V., & Deci, E. (1983). *Management and Motivation*. Penguin.

Weaver. R., & Rockman B. (1993). *Do Institutions Matter? Government Capabilities in the United States and Abroad.* Washington DC: The Brookings Institution.

World Wide Web Consortium. (2012). Retrieved from http://www.w3.org/XML/.

Appendices

Appendix A: Questionnaire

A MODEL FOR IMPROVING INTEROPERABILITY OF HEALTHCARE SYSTEMS

Dear Respondent,

I am a Masters student in the Faculty of Information Technology, Strathmore University conducting a research entitled **A MODEL FOR IMPROVING INTEROPERABILITY OF HEALTHCARE SYSTEMS IN A DISTRIBUTED ENVIRONMENT**.

As part of my research, you are hereby selected to aid in achieving the objectives of this study. I hereby request you to fill in the questionnaire below. Please do note that the information furnished here is purely for academic purposes and thus its confidentiality shall be safeguarded.

Kind Regards, Benson Ogutu.

SECTION A: Respondent Details (ALL)

1. **Choose your age group:**

☐ 20 - 30 years

☐ 31 years - 40 years

☐ 40 years - 50 years

☐ Over 50 years

2. **On a scale of 1-5 (5 being the highest) rate your experience in using a Healthcare Management System (HMS): (Healthcare Staff Only) Choose one:**

☐ 5

☐ 4

☐ 3

☐ 2

□ 1

SECTION B: FACTORS THAT AFFECT INTEROPERABILITY IN HEALTHCARE SYSTEMS (Healthcare Staff Only)

1. **What information do you normally access from the Healthcare System? Choose from the list below:**

   a. Patient's details

   b. Patient's Diagnostic information

   c. Patient's Prescription information

   d. Patient's Financial information

   e. Other. Please state ____

2. **From the factors listed below, which ones do inform you on the need of an interoperable healthcare system? Select from the factor identified below: (Healthcare Staff Only)**

   a. Ease of access

   b. Data portability

   c. Data confidentiality, integrity and security

   d. Capture of different data formats

   e. File sharing

   f. Cost reduction

   g. Scalability of systems

SECTION C: INTEROPERABILITY HEALTHCARE STANDARDS AND ARCHITECTURES FOR A DISTRIBUTED HEALTHCARE ENVIRONMENT

Standards: (Healthcare Staff Only)

1. **On a scale of 1 – 5 (where 5 is the highest and 1 the least), how would you rate the level of standardization of the healthcare semantic standards?**

   5 – High standardization

   4 – Medium Standardization

   3 – Average

   2 – Low standardization

   1 – Very Low standardization

2. **On a scale of 1 – 5 (where 5 is the highest and 1 the least), how would you rate the level of standardization of the healthcare syntax standards? (Healthcare Staff Only)**

   5 – High standardization

   4 – Medium Standardization

   3 – Average

   2 – Low standardization

   1 – Very Low standardization

Architectures:

3. **What are the architectures used in developing your distributed healthcare systems? Select from the options below: (Healthcare ICT Staff Only)**

   a. Master-slave architecture

   b. Two-tier client–server architecture

   c. Multitier client–server architecture

d. Distributed component architecture

e. Peer-to-peer architecture

Patient:

What are the challenges experiences when being referred from one hospital to the next?

____

____

____

____

If a system for allowing information sharing between 2 healthcare systems was developed, would you allow your health record to be shared across?

a. Yes

b. No. If No, Why? ____

What information would you want shared across?

____

____

____

____

____

Appendix B: User Acceptance Questionnaire

Having seen how the API works, would you recommend its implementation or not?

☐ Yes. I would recommend its implementation.

☐ No. I would not

On a scale of 1 – 5 with 1 being the least and 4 being most, rate the usability of the API:

☐ 5

☐ 4

☐ 3

☐ 2

☐ 1

What challenges do you foresee with regard to the actual implementation and acceptance of the system in the country?

___

___

___

___

___