UNIVERSITY OF NAIROBI

SCHOOL OF COMPUTING AND INFORMATICS

SYSTEM INTEROPERABILITY WEB DATA-EXCHANGE SERVICE BUS FOR INTEGRATING HEALTH INFORMATION SYSTEMS

BY

JONATHAN MWAI

Registration Number: P58/64026/2011

Submitted for the partial fulfillment for the requirements of Masters of Science in Computer Science

JUNE 2016

DECLARATION

I Jonathan Mwai declare that this project research, is my own original work and has not been presented anywhere for the purpose of an academic award.

Signature: _____

Jonathan Mwai N

P58/64026/2011

Supervisor:

Professor Peter Wagacha Waiganjo

University of Nairobi, School of Computing and Informatics

Signature: _____

ACKNOWLEDGEMENT

I would like to gratefully take this opportunity to thank my Almighty God for his care, grace and good health He offered to me throughout the Master of Science course.

I would also like to express my heartfelt gratitude to Professor Peter Wagacha (Supervisor) for your insights, exemplary guidance, valuable feedback and constant encouragement throughout the duration of the project have been outstanding, Thank you.

ABSTRACT

Given the need to manage increasing information and knowledge within Kenya's health sector, developments in information technology have become more crucial to meet the required demand. Healthcare providers especially government hospitals have made huge investments towards infrastructure improvement and development/purchase of new required health management information systems. Despite the government and private healthcare providers having invested considerably more in the acquisition of various systems, these systems cannot generate expected outcomes if not integrated to achieve common national goals like measuring of health service delivery, morbidity control, etc. A fundamental concern in health management is the integration of health information across distributed, heterogeneous and disparate information systems. Lack of interoperable health systems is one of the major barriers to the use of health information.

DHIS 2 is a national health information system (HIS) that was deployed in the country by the government of Kenya in 2010, its function is monitoring health, and evaluating and improving the delivery of health-care services and programs in the country. It is also used for reporting, analysis and dissemination of health data obtained from health facilities and hospitals nationwide. Thus the processed and analysed information from DHIS2 is used for national Health Decision Making by the government health managers, stakeholders and donors for resource allocation. However untimely, incomplete and inaccurate data are the main challenges that have faced the national HIS since the health reporting is paper based dominated and also due to lack of integration between the fragmented health information systems that could submit health data electronically and automatically without using the paper reporting tools.

In this work through literature review, the research is based on a need for a software infrastructure that will enable integration of the national DHIS2 with the other existing disparate HISs used in health facilities in order to promote interoperability between the systems. Service-Oriented Computing (SOC) is a new computing paradigm that utilizes services as the basic constructs in development of rapid, low-cost and easy composition of distributed system. Service Oriented Architecture (SOA) was adopted as the application framework in designing, building and implementing the service-based solution. Enterprise Service Bus (ESB) a layer of middleware through which a set of core (reusable) services are made widely available, this approach was used in development of a web data exchange service bus for integrating the HISs thus facilitating interoperability of the different systems across platforms, enhance communication and data exchange.

After the development and evaluation of the Web Data Exchange Bus that enabled an instance of DHIS2 Kenya to interoperate with other Partner-HISs, it is evident that SOA enabled-infrastructure is the most ideal method of integrating systems compared to others methods like point-to-point integrations (through Application Programming Interfaces - APIs) which provides no flexibility of systems changes without impacting on each other. The study demonstrated that Service Bus can be used to integrate new Web Service (WS) based systems with legacy systems that do not have APIs functionalities.

TABLE OF CONTENT

Declaration ii
Acknowledgement iii
Abstract iv
List of Figures viii
List of Abbreviations ix
Key Terminologies xi
1.0 CHAPTER ONE: INTRODUCTION 1
1.1 Background Information 1
1.2 Problem Statement 2
1.3 Purpose of the Project 3
1.4 Objectives 3
1.5 Significance of the Study 3
1.6 Research Outcomes 3
1.7 Assumptions and Limitations 4
2.0 CHAPTER TWO: LITERATURE REVIEW 5
2.1 Types of interoperability pertinent to health 5
2.2 Levels of interoperability 6
2.3 Interoperability standards 7
2.3.1 Categories of standards used in healthcare: 7
2.3.2 Standards development organizations 9
2.4 Health Interoperability Standards 10
2.4.1 Health Level 7 (HL7) 11
2.4.2 Reference Information Model (RIM) 12
2.4.3 What are Services 16
2.4.4 Types of Service 17
2.4.5 What is SOA? 17
2.4.6Basics of SOA 18
2.5Service-Oriented Architecture (SOA)[11] 18
2.5.1SOA Model 18
2.5.2Advantages of SOA 20
2.6 Web Service 21
2.6.1Components of Web Services [12] 22
2.6.2Security Concerns 23
2.6.3 How Web Services Work[12] 24
2.7Evolution of Enterprise Application Integration Architectures 27
2.7.1Point to Point Topology 27
2.7.2 Hub and Spoke Topology 28
2.7.3 Bus Topology 29
2.8 Enterprise Service Bus 30
2.9 Overview of Web Services and their applications 34
Related works 35
2.9.1 A SOA based architecture to promote ubiquity and interoperability among HISs 35
2.9.2 Web Service-Based Integrated Healthcare Information Systems (WSIHIS) 36
2.9.3 Artemis 38
3.0 CHAPTER THREE: METHODOLOGY 40
3.1 Introduction 40
3.2Web services development methodology 41

4.0 CHAPTER FOUR: SYSTEM ANALYSIS, DESIGN & IMPLEMENTATION..... 44
4.1Software Tool used for development 44
4.1.2User requirements 45
4.1.3 Use case diagram 46
4.1.4 The Integration Approach 48
4.1.5The Interoperability Process of the Web Service Data Exchange Bus 50
4.1.6 The interoperability process consist of following main stages ;- 51
4.1.7Data Exchange Service Bus Data Flow 52
4.1.8The Web Data Exchange Bus Activity Diagram 53
4.1.9Canonical Data Format 54
4.2.0Non-Functional Requirements 55
5.0 CHAPTER FIVE: RESULTS AND DISCUSSION 63
5.1 There are desirable characteristics in interoperability achieved the system. 63
5.2 Comparison between Web services and API Integrating Approaches 64
5.3 Conclusion 65
5.4 Recommendation for Future Work 66
References 67
Appendices 70
Sample of MOH 711 Register Reporting Form 70
Sample code 72

LIST OF TABLES

*Table 1:*Comparison between Web services and API integrating approaches 65

LIST OF FIGURES

Figure 1: HL7 layers of data transmission 12
Figure 2: HL7 v3 information refinement process 14
Figure 3: SOA model 19
Figure 4: Web Services – the SOA implementation 22
Figure 5: Web Service Technologies 25
Figure 6: Interaction between applications/consumers and Web services. 26
Figure 7: Point To Point Topology 27
Figure 8: Hub and Spoke Topology 28
Figure 9: Bus Topology 29
Figure 10: Enterprise Service Bus Architecture[30] 32
*Figure 11:*Web Services Development Workflows 40
*Figure 12:*Web Service Development Methodology (extending Agile Methodology.) 41
*Figure13:*Diagram of the Web Data Exchange Service Bus for HIS 45
Figure 14: Use Case of the Web Data Exchange Service prototype 47
Figure 15: The layer architecture for health information systems 48
Figure 16: Usage of the Interoperability Layer by the Health Information Systems 50
*Figure 17:*Interoperability process 50
*Figure 18:*Data Exchange Service Bus Data Flow 52
*Figure 19:*Data Exchange Service Bus Activity Diagram 53
Figure 20: Transformation process 55
*Figure 21:*Login user interface of the prototype - BUS 57
*Figure 22:*Login user interface to the recipient system - DHIS2 57
Figure 23:Interface of the prototype after gaining connection to Service ProviderDHIS2 58
*Figure 24:*Interface of prototype and listing health facilities 58
Figure 25: Interface affording user options for sourcing data 59
*Figure 26:*User options to load a text file with the values to be consumed by another HIS. . 60
*Figure 27:*Harvested HIS data values but not mapped them to DHIS2 data elements 60
*Figure 28:*Interface for mapping data values to the data elements. 61
*Figure 29:*Specify location/ip-address of the remote host, database connection credentials . 61
Figure 30:Before data was received by the recipient DHIS2 system 62
*Figure 31:*Data received by recipient system DHIS2 62
Figure 321: Sample of MOH 711 Register Reporting Form 72

LIST OF ABBREVIATIONS

ANSI - American National Standards Institute

API - Application Programming Interface

CDA - Clinical Document Architecture

CCOW - Clinical Context Object Workgroup

CEN - ComitéEuropéen de Normalisation

CHRIO - County Health Records Information Officer

CORBA - Common Object Request Broker Architecture

D-MIM - Domain Message Information Model

DHIS2 - District Hospital Information System2

DICOM - Digital Imaging and Communications in Medicine

EAI - Enterprise Application Integration

EMRS - Electronic Medical Record System

EHR - Electronic Health Record

ESB - Electronic Service Bus

FTP – File Transfer Protocol

HIS - Health Information System

HISA - Healthcare Information Systems Architecture

HL7 - Health Level Seven

HTTP - Hypertext Transfer Protocol

HTTP - Hypertext Transfer Protocol Secure

HMD - Hierarchical Message Description

HMIS - Health Management Information System

IEEE - Institute of Electrical and Electronic Engineering

IETF - Internet Engineering Task Force

ISO - International Standards Organization

JMS - Java Message Service

JSON - JavaScript Object Notation

LOINC -Logical Observation Identifiers Names and Codes

MOH - Ministry of Health

RMIM - Refined Message Information Model

RMI - Remote Method Invocation

RIM - Reference Information Model

SDO - Standards Development Organizations

SNOMED CT -Systematized Nomenclature of Medicine Clinical Terms

SOAP - Simple Object Access Protocol

SSL - Secure Socket Layer

TCP - Transmission Control Protocol

MOM - Message Oriented Middleware

UDDI - Universal Description, Discovery, and Integration

WS - Web Service

WSDL - Web Services Description Language

WWW - World Wide Web

W3C - World Wide Web Consortium

XDR - Cross-Enterprise Document Reliable Interchange

XML - eXtensible Markup Language

KEY TERMINOLOGIES

MOH 711

It is a register designed to capture service delivery data provided to persons who visited health facilities to consult, counseled, tested or treated etc. on Integrated RH, HIVAIDS, Malaria, TB& Nutrition on a monthly basis. The register is designed to aggregate units of data elements used for reporting. (*Sample MOH 711 register attached at appendices*)

County Health Records Information Officer (CHRIO)

Health or hospital officers deployed by the government at County level hospitals who are custodian of records information in the facility and are also responsible of receiving reporting data from all the health facilities in that specific sub-county and entering the data into DHIS2.

1.0 CHAPTER ONE: INTRODUCTION

1.1 Background Information

With the emerging of IT technologies and increasing demands for computerizing medical-related information, health institutions have developed systems to manage and process the large amount of medical information. Exchange of clinical information and medical records amongst health providers would provide safe and reliable healthcare this depends on access to, and the use of, information that is accurate, valid, reliable, timely, relevant, legible and complete.

For example, when giving a patient a drug, a nurse needs to be sure that they are administering the appropriate dose of the correct drug to the right patient and that the patient is not allergic to it. Similarly, lack of up-to-date information can lead to the unnecessary duplication of tests – if critical diagnostic results are missing or overlooked, tests have be repeated unnecessarily and, at best, appropriate treatment is delayed or at worst not given. Health information has a key role to play in healthcare planning decisions – where to locate a new service, whether or not to introduce a new national screening programme and decisions on best value for money in health and social care provision.

In order to achieve information exchange and medical knowledge sharing, hospitals and healthcare providers have intended to integrate their systems' functions and data. This has raised some concerns, such as, data security, data transmission, network limitation and so on. Among these issues, the issue of system and data interoperability are most obvious barrier for the integration of functions and data of different systems.

Nowadays health information systems have been developed using different languages (e.g. Java, Visual Basic, C#, etc.) different system platforms (e.g. Microsoft Windows, Linux, etc.) and Database Management Systems (e.g. Microsoft SQL server, Oracle, PostgreSQL, Sybase, etc), these differences between systems are the major factors leading to system interoperability where the valuable data stored in disparate systems cannot be exchanged across platforms and be used by any health stakeholder who needs the information.

The need of exchange of electronic health information has been felt in whole sector of healthcare across the world and this has prompted international organizations to set standards for all aspects of health information to allow seamless sharing of the information across organizational boundaries, the standards are commonly referred to as interoperability standards. The Interoperability standards that have been developed provide a standardized

approach to facilitate flawless exchange of information between health information systems. Different organizations have developed models, frameworks, service and information representations and approaches in-order to address interoperability issues.

Web services have emerged as the next generation of integration technology. Based on open standards, the Web services technology allows any piece of software to communicate with each other in a standardized XML messaging systems. It solves and eliminates problems of DCOM/CORBA distributed middleware technologies. Microsoft .Net framework and C language are important part of solution for the interoperability as they are language neutral. The main objective of the proposed project is to research on the issues of system and language interoperability and develop a Web Data-exchange service for Integrated Health Information Systems.

1.2 Problem Statement

Currently the District Health Information System (DHIS2) deployed nationally at county level, Health Management Information System (HMIS) deployed in most health facilities, Electronic Medical Record System (EMRS) and others are health information systems that contain very valuable patients' medical data that is accessed by health providers at health facilities. Due to the fact that the information systems used by the health institutions are independent there is difficulty in accessing the required patients' prescriptions, past health history for the purposes of appropriate treatment especially if the patient is seeking treatment in a different health facility from the he/she is used to. Lack of the systems interoperability has hindered ease of access to medical data/records leading to risks of erroneous treatments, inefficient and poor quality healthcare.

For a better decision-making process, especially in a critical area as healthcare, fast and reliable access of a patient's medical history is of utmost importance (Maass et al, 2008; Revere et al, 2007), even though the information is located in a different information system geographically distant from de information system in use in a given situation.

There is an urgent need for the creation of integration mechanisms among the different health information systems that exists today in different healthcare units, like what happens in other areas of activity (Manpaa et al, 2009).

An environment that allows exchange and sharing of the medical records, healthcare knowledge and information would enhance efficiency and quality of healthcare in kenya.

1.3 Purpose of the Project

This research will explore a web data exchange service interoperability system to enhance quality, effectiveness and safety in healthcare by enabling sharing for data and information within health information. To provide cost-effective health information system integration solution, this research proposes implementation of a prototype that helps us to share not only the data in these HISs but also the data management platform in order to promote efficiency in decision making by hospital managements. This research proposal will evaluate the Web Service approach as an appropriate methodology to achieve health information system interoperability.

1.4 Objectives

i. Identify the relevant existing data interoperability technologies and standards.

ii. Identify the problems and the solutions related to data exchange between disparate Hospital Information Systems (HISs).

iii. To understand how a data sharing platform/interoperability system can be implemented

iv. To develop a data exchange prototype using Web Services technologies that will enhance data exchange between the major Health Information Systems and the disparate HISs.

1.5 Significance of the Study

This study proposes developing of a prototype that helps to exchange and share the data and information in HISs. For health professionals and patients will benefit from efficiency in access of data anytime and anywhere e.g. the medical records and past treatments of patients. The study will improve access and availability to health record data and health information anytime, anywhere, thus enhancing quality and safety of care by improving data exchange, the quality of data flow and access to information by health professionals thereby potentially reducing errors.

1.6 Research Outcomes

The deliverables for this study will be a prototype that will help the healthcare professionals to access valuable medical data and information from other crucial health information systems. After this study, the deliverables will include contribution of knowledge and a well published paper.

1.7 Assumptions and Limitations

This study assumes that there is a functional database in each and every healthcare institution that can be queried to find records and to extract information. This research study will also assume that there are computers and the health information systems in all health which are internetworked via intranets and Internet.

This study assumes that the HL7 version 3 standards have been adopted for representation and recording of medical content in electronic documents across all information systems nationally. This proposal involves development of a prototype system that implements the proposed solution to demonstrate its application in a real-world setting.

There are numerous hospitals (private and public) that have developed proprietary Information Systems that are dependent, it is very difficult to realize interconnection of the all these systems. We will limit our research on sharing and exchange of data between the three major health information systems namely DHIS2, HMIS and EMRS

2.0 CHAPTER TWO: LITERATURE REVIEW

Interoperability the ability of two or more systems or components to exchange information and to use the information that has been exchanged (IEEE Glossary).[1] Being able to accomplish end-user applications using different types of computer systems, operating systems, and application software, interconnected by different types of local and wide area networks(O'Brien J, Marakas G)[2]

2.1 Types of interoperability pertinent to health

Interoperability issues can be considered from three different viewpoints to maximize business benefit:[3]

a. Technical interoperability is the exchange of data between computer system A and computer system B. The computers do not know about the meaning of what is exchanged. For example, emails transmitted from one computer to another generally contain content information that is not understood by the sending or receiving computer.

b. Semantic interoperability guarantees that computer system A and computer system B understand the meaning of data in the same way and use and interpret the data that is exchanged. Semantic interoperability is central to healthcare interoperability. For example, a laboratory information system transmits results to a practice management system at a GP practice. The practice management system recognizes the structure, format, units and meaning of the result sent by the laboratory system. In order to achieve this, both systems use a common terminology or language to communicate.

c. Process interoperability incorporates business processes. It is important that business processes also interoperate and the people involved share a common understanding to enable computer system A and computer system B to work together. For example, healthcare professionals must standardize business rules to ensure that health information is recorded in a uniform and timely manner such that the transfer of information between systems is consistent and complete.[3]

To support interoperability between systems and meaningful sharing of data, health information standards must cover both the syntax (structure) and semantics (meaning) of the data exchanged. Interoperability standards are guidelines that technology developers can use to develop health information systems that will be inherently compatible with other systems adhering to these same standards.[4]

The use of interoperability standards delivers key benefits in a number of areas: standards enable and support health service improvements, deliver economic benefits and, most importantly, result in benefits for individuals through safety improvements in service delivery. In the area of implementation, standards acts as the middle ground where coordination between different software systems is needed. [5]

Interoperability greatly benefits all involved in the delivery and receipt of healthcare:

i. Patients can benefit from enhanced quality and safety of treatments received, delivery of healthcare when and where it is required.
ii. The electronic transfer of prescriptions (ETP) can be enabled through interoperability of pharmacy systems with primary care information systems facilitating a reduction in the potential for harmful drug interactions and transcription errors.
iii. Healthcare professionals can potentially improve the quality and safety of the care they provide through strengthened coordination across the various points of care delivery.
iv. Individuals and healthcare professionals can benefit from efficiency gains due to a reduction in duplication of data entry, such as recording of the same demographic information at multiple locations.
v. Interoperability standards can benefit the software industry by enabling a single market for digital healthcare, thereby reducing the cost of developing health information systems and opening up competition in the market.
vi. Efficiency gains brought about by the implementation of healthcare interoperability standards can benefit the health officers and patients facilitating faster access to care, diagnosis and treatment of disease, thereby reducing costs significantly.[3]

2.2 Levels of interoperability

There are four levels of interoperability, each demonstrating a level of sophistication and standardization of health information interoperability:[6]

1. Non-electronic information – there is minimal use of technology to share data and most health information is recorded and shared on paper. For example, referral from primary care to secondary care by paper-based referral letter sent via standard postal service.

2. Machine transportable information – transmission of non-standardized data using basic information technology. This data cannot be electronically manipulated. For example, sharing of paper-based health information via fax or email attachment.

3. Machine organisable information – transmission of structured electronic messages containing non-standardized data. This means that information can be shared electronically. However, an interface is required between one or more systems to translate the data from the structure used by the sending system to the structure used by the receiving system.

4. Machine interpretable information – transmission of structured messages containing standardized and coded data. This means that systems exchange health information electronically using a format and vocabulary that is readable and interpretable by the receiver without the requirement for an interface to decode the information. For example, a discharge summary is transmitted electronically from the hospital information system to the primary care electronic record of the patient in a structured and coded format that is used by both systems, such as HL7 Clinical Document Architecture (CDA) and SNOMED CT.[6]

2.3 Interoperability standards

Standards for healthcare interoperability exist to allow health information systems to communicate in the same way across system, organizational, regional and national boundaries. interoperability can also be categorized into various levels, each indicating a level of complexity of health information exchange. In order to facilitate complex levels of interoperability, a number of standards development organizations (SDOs) exist. These organizations develop adoptable standards for the various types or categories of interoperability, many of which can operate in tandem to allow functional and semantic interoperability.

2.3.1 Categories of standards used in healthcare:

a. **Messaging standards** – messaging standards outline the structure, content and data requirements of electronic messages to enable the effective and accurate sharing of information. The term 'message' refers to a unit of information that is sent from one system to another, such as between a laboratory information system and a GP's clinical information system. Examples of messaging standards include HL7 v2.x for administrative data and Digital Imaging and Communications in Medicine (DICOM) for radiology images. [3]

b. **Terminology standards** – terminology standards provide specific codes for terminologies and classifications for clinical concepts such as diseases and medications. Terminology systems assign a unique code or value to a specific disease or entity, for example, the ICPC-2 code for 'asthma' is R96.

Terminologies are used primarily to capture clinical information at the point of care. As such, they are highly detailed, have predefined relationships and are fine grained.

c. **Document standards** – document standards indicate the type of information included in a document and also the location of the information. Examples of document standards include the paper-based Subjective, Objective, Assessment, Plan (SOAP) standard and also HL7 Clinical Document Architecture (CDA) for electronic sharing of clinical documents. HL7 have developed document-standard specifications for a continuity of care document (HL7 CCD) and a discharge summary (HL7 DS).

d. **Conceptual standards** – conceptual standards allow the transmission of information between systems without any loss of the meaning or context of that information. For example, the HL7 Reference Information Model (RIM) provides a framework for describing health information and the context around it, i.e. who, what, when, where and how.

e. **Application standards** – application standards determine the implementation of business rules for software systems to interact with each other. For example, application standards can allow a single user to log in to multiple information systems in one environment allowing efficient access to the required health information. This can facilitate the simultaneous viewing of health information across multiple databases that are not electronically integrated.

f. **Architecture standards** – architecture standards define a generic model for health information systems. They allow the integration of health information systems by providing guidance to aid the planning and design of new systems and also the integration of existing systems. This is achieved by defining common data elements and business logic between systems. For example, the CEN standard ENV12967 (Healthcare Information Systems Architecture or HISA) provides an open architecture that is independent of technical specifications and applications. This standard enables integration of common data and business logic between systems, which is achieved via a middleware § layer allowing
information exchange between different systems.[3]

2.3.2 Standards development organizations

There are a number of international standards development organizations (SDOs) that have developed interoperability standards to facilitate the exchange of health information that have achieved widespread adoption around the world..

a. International Standards Organization (ISO)

ISO/TC215, was established for the area of health informatics with the scope of:
standardization in the field of information for health, and health information and communications technology to achieve compatibility and interoperability between independent systems. ISO/TC215 collaborates with a number of other SDOs including ComitéEuropéen de Normalisation (CEN) and HL7. [3]

b. Comité Européen de Normalisation (CEN)

CEN, or the European Committee for Standardization, is involved in developing multidisciplinary standards including standards for healthcare systems and interoperability. TC 251 is the health informatics technical committee in CEN with responsibility for publishing standards addressing aspects of health information representation including messaging, electronic health records and eHealth initiatives.

c. Health Level Seven (HL7)

The organization is an SDO accredited by the American National Standards Institute (ANSI) with the purpose of developing and publishing healthcare-specific standards. It publishes messaging standards for healthcare interoperability that aim to enhance care delivery, knowledge transfer and optimize workflow. HL7 products include HL7 version 2.x (v2.x), HL7 version 3 (v3) messaging standard, Clinical Document Architecture (CDA), Clinical Context Object Workgroup (CCOW) and Arden Syntax.

Other major organizations that have been mandated to develop healthcare standards are **OpenEHR**and Integrating the Healthcare Enterprise (**IHE**).

Internationally HL7v3 is by far the most widely used standard for exchanging healthcare messages. OpenEHR and CEN13606 are similar but neither has reached critical mass in terms of adoption.

2.4 Health Interoperability Standards

i. Health Level 7 (HL7)

HL7 is a collection of message formats and related clinical standards that define an ideal presentation of clinical information, and together the standards provide a data exchange framework. HL7 is a standard for healthcare specific data exchange between computer applications. The name comes from "Health Level 7" (top layer of the Open Systems Interconnection layer protocol for the health environment.

ii. Continuity of Care Record (CCR)

CCR is an XML - based standard for the movement of "documents" between clinical applications. Furthermore, it responds to the need to organize and make transportable a set of basic information about a patient's health care that is accessible to clinicians & patients.

iii. Clinical Context Object Workgroup (CCOW)

CCOW is an HL7 standard protocol designed to enable disparate applications to synchronize in real-time and at the user-interface level. It is vendor independent and allows applications to present information at the desktop and/or portal level in a unified way.

iv. Clinical Document Architecture (CDA) HL7

CDA uses XML for encoding documents and breaks down the document in generic, unnamed, and non-templated sections. Documents may include discharge summaries, progress notes, history and physical reports, prior lab results, etc. HL7's CDA defines a very generic structure for delivering "any document" between systems. CDA was previously known as the Patient Record Architecture (PRA).

v. Digital Imaging and Communications in Medicine Committee (DICOM)

DICOM is a standard for handling, storing, printing, and transmitting information in medical imaging. It includes a file format definition and a network communications protocol. The communication protocol is an application protocol that uses TCP/IP to communicate between systems. DICOM files can be exchanged between two systems that is capable of receiving image and patient data in DICOM format.

vi. **Cross-Enterprise Document Reliable Interchange (XDR)**

XDR used the exchange of health documents between health enterprises using a web-based, point-to-point push network communication, permitting direct interchange between EHRs, PHRs and other systems without the need for a document repository.

Example: A nurse at Hospital A enters a patient's information in the local EHR, and then sends the CCD (a clinical document exchange standard) directly to Hospital B's system.

vii. **Logical Observation Identifiers Names and Codes (LOINC)**

LOINC applies universal code names and identifiers to medical terminology related to the EHR and assists in the electronic exchange and gathering of clinical results (laboratory tests, clinical observations, outcomes management, research) [8]

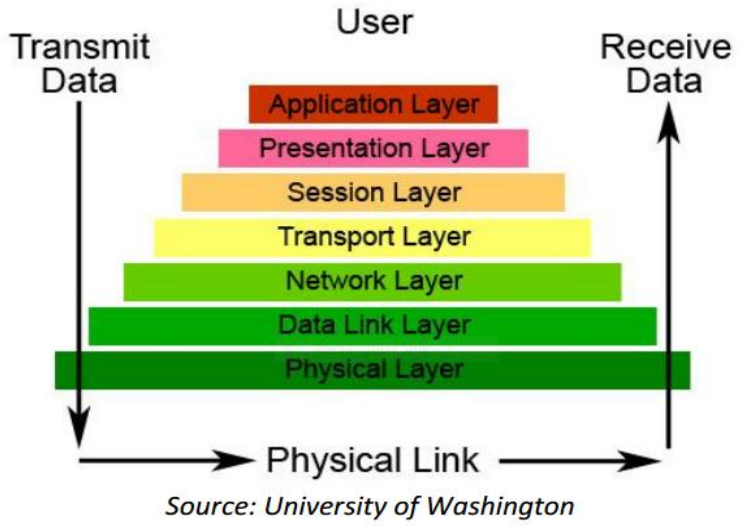
2.4.1 Health Level 7 (HL7)

HL7 Provides standards for data exchange to allow interoperability between healthcare information systems and focuses on the clinical and administrative data. Key goal is of***syntactic and semantic interoperability*** . Over 90% of US hospitals have implemented some version of HL7 messages.

i. In healthcare, HL7 has been used since the 80s (pre internet era)
ii. It used the Unix pipe (|) as a data delimiter. The current internet standard for document markup is XML, which uses "<>" as a data delimiter.
iii. The data delimiters are used to structure the data
iv. Without a data dictionary to translate the contents of the delimiters, the data remains meaningless. While there are many attempts at creating data dictionaries to associate with these data packaging mechanisms, none have been practical to implement.
v. This has perpetuated the ongoing data inability to exchange data with meaning. [8]

What does HL7 Mean

"Level seven" refers to the highest level of the International Standards Organization's (ISO) communications model for Open Systems Interconnection (OSI) - the application level. The application level addresses definition of the data to be exchanged, the timing of the interchange, and the communication of certain errors to the application.



[8]

Figure 1: **HL7 layers of data transmission**

The goal of HL7

i. HL7 V3 is a standard for exchanging messages among information systems in healthcare
ii. To accomplish this, HL7 uses an object-oriented development methodology and is based on a Reference Information Model (RIM) to create messages
iii. The goal of HL7 is semantic interoperability

Standards the HL7 has developed

- Conceptual standards (e.g., HL7 RIM)
- Messaging standards (e.g., HL7 v2.x and v3.0)
- Application standards (e.g., HL7 CCOW)
- Document standards (e.g., HL7 CDA)[8]

2.4.2 **Reference Information Model (RIM)**

RIM is the cornerstone of the HL7 Version 3 development process. It is the fundamental model from which all v3 messages are derived is referred to as the Reference Information Model (RIM)

- RIM follows object oriented methodology based on a UML model
- The RIM is a generic, abstract model that expresses the information content of all health areas
- Defines all the information from which the data content of HL7 messages are drawn

- Forms a shared view of the healthcare domain and is used across all HL7 messages independent of message structure
- RIM model is color coded based on the backbone class
- It is based on classes and their structural attributes[9]

RIM expresses the data content needed in a specific clinical or administrative context and provides an explicit representation of the semantic and lexical connections that exist between the information carried in the fields of HL7 messages.

RIM is an information model An information model is a structured specification of the information in a specific domain of interest.

An Information Model consists of

i. classes, their attributes, and relationships between the classes;
ii. data types for all attributes and vocabulary domains for coded attributes;
iii. state transition models for some classes.

HL7 information models are based upon the Unified Modeling Language (UML)*, and may be represented graphically using the UML style.

UML is a modeling language based on object-oriented modeling methods

HL7 version 3 (HL7 v3) uses

Reference Information Model (**RIM**), an object model that is a large class diagram representation of the clinical data and identifies the life cycle of events that a message will carry, and applies object-oriented development methodology on RIM and its extensions to create messages. In the following, two HL7 concepts, i.e.,

a. **Message Structure** and
b. **Refinement Process**

are briefly discussed; these concepts are the basis of our translation process [4, 10].

Message Structure. At the highest level, an HL7 version

3 message is composed of two parts:

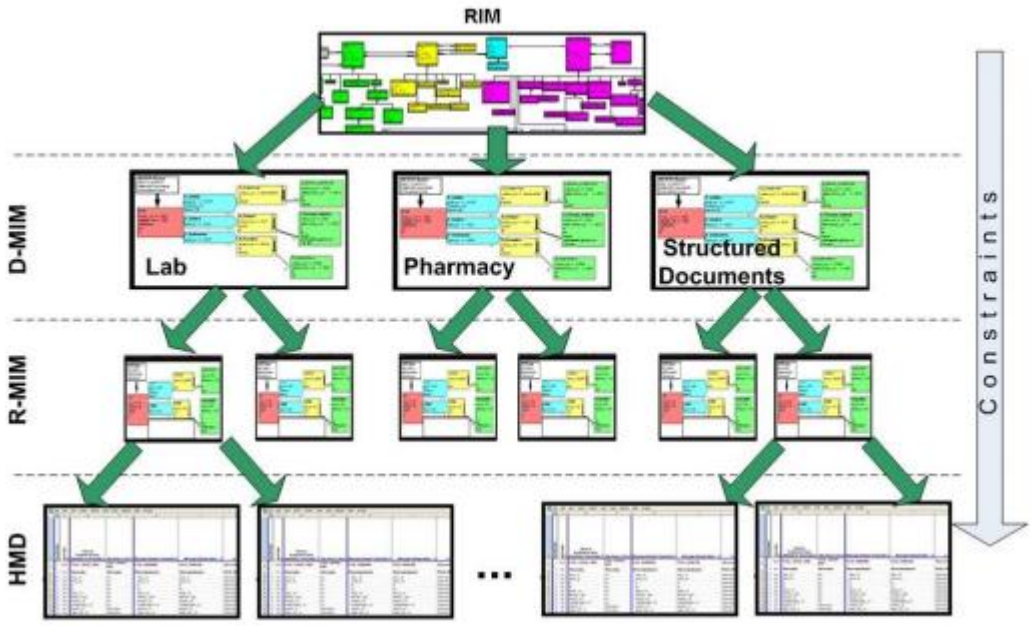


Figure 2: **HL7 v3 information refinement process**

• *HL7 Transmission Wrapper* includes information needed by a sending application or message handling service to package and route the v3 messages to the designated receiving applications or message handling services.

• *HL7 Transmission Content* contains core data at-tributes for the message such as a prescription order or dispense event. It also includes information about the business event that initiated the sending of this message, which sent it and other associated business information.

Refinement process - HL7 methodology uses RIM, HL7-specified Vocabulary Domains, and version 3 Data Type Specification as its starting point. It then establishes

the rules for refining these base standards to arrive at the in-formation structures that specify Message Types and equivalent structures in v3.

The strategy for development of version 3 messages and related information structures is based upon the consistent application of constraints to a pair of base specifications,

i.e., HL7 RIM and HL7 Vocabulary Domains, and upon the extension of those specifications to create representations constrained to address a specific healthcare requirement. Figure 1 shows the refinement process specified in HL7 methodology, where the different parts are discussed below.

•*Domain Message Information Model (DMIM)* is a subset of the RIM that includes a fully expanded set of class clones, attributes and relationships that are used to create messages for any particular domain.

• *Refined Message Information Model (R-MIM)*is used to express the information content for one or more messages within a domain. Each R-MIM is a subset of
the D-MIM and contains only those classes, attributes and associations required to compose the set of messages.

•*Hierarchical Message Description (HMD)* is a tabular representation of the sequence of elements (i.e., classes, attributes and associations) represented in an R-MIM. Each HMD produces a single base message template from which the specific message types are drawn.

•*Message Type* represents a unique set of constraints that are presented in both grid and table view as well as an Excel spreadsheet.

HL7 version 3 will be used to achieve Semantic Interoperability and Syntactic Interoperability. The use of HL7 v3 open standards and RIM model for sharing of the data information model allows the overcoming of the Syntactic interoperability.

Technical interoperability is obtained by use of standard communication protocols (HTTP, SOAP etc.) and by definition of standardized communication interfaces (WSDL services). The web services offer the promises and hopes of integrating disparate applications in a seamless fashion and solve the integration problems for the business enterprises. Traditionally, solutions of interoperability normally involve developing middleware applications to communicate the non-interoperable applications using the messages, of which the technology is also named as the distributed middleware technology. CORBA and DCOM are two most typical distributed middleware technologies. Nevertheless, CORBA or DCOM is a fairly complex, a task requiring special expertise. Quite often, the achievement of a good interoperability strategy is significantly constrained by many implementation restrictions in CORBA or DCOM[9]

.Web services have emerged as the next generation of integration technology. Based on open standards, the Web services technology allows any piece of software to communicate with each other in a standardized XML messaging systems Web services are modular self-describing and self-contained applications that can be published, located and dynamically invoked across the Web.

2.4.3 What are Services

Service is a self-contained, platform- independent computational element that support rapid, low-cost and easy composition of loosely coupled distributed software applications **[10]**. The functionality provided by a service can range from answering simple requests to executing sophisticated processes requiring peer-to-peer relationships between multiple layers of service consumers and providers.

Services help integrate applications that were not written with the intent to be easily integrated with other applications and define architectures and techniques to build new functionality while integrating existing application functionality. Service-based applications are developed as independent sets of interacting services offering well-defined interfaces to their potential users. This is achieved without the necessity for tight coupling of applications between transacting partners, or for pre-determined agreements to be put into place before the use of an offered service is allowed.[20]

Some Characteristics of Services:

i. *Supports open standards for integration* - Although proprietary integration mechanisms may be offered by the SOA infrastructure, SOA's should be based on open standards. Open standards ensure the broadest integration compatibility opportunities.

ii. *Loose coupling* -The consumer of the service is required to provide only the stated data on the interface definition, and to expect only the specified results on the interface definition. The service is capable of handling all processing (including exception processing).

iii. *Stateless* - The service does not maintain state between invocations. It takes the parameters provided, performs the defined function, and returns the expected result. If a transaction is involved, the transaction is committed and the data is saved to the database.

iv. *Location agnostic/transparent* - Users of the service do not need to worry about the implementation details for accessing the service. The SOA infrastructure will provide standardized access mechanisms with service-level agreements.

v. Have a network-addressable interface

vi. Have a coarse-grained interfaces

vii. Self-contained and modular

2.4.4 Types of Service

Topologically, services can come in two flavors: (simple) informational or complex services. Each of these two models exhibits several important distinguishing characteristics that are briefly described below.

a. **Informational services** are services of relatively simple nature. They either provide access to content interacting with an end-user by means of simple request/response sequences, or alternatively they may expose back-end business applications to other applications. Informational services can be of various kinds:

   i. *Pure content services*, which give programmatic access to content such as simple financial information, stock quote information, design information, news items and so on.

   ii. Seamless aggregation services, which provide seamless aggregation of information across disparate systems and information sources including back-end systems.logistic services, where automated services are the actual front-ends to fairly complex physical organizational business processes.

   iii. *Information syndication services*, which are services offered by a third-party and run the whole range from commerce-enabling services, such as logistics, payment, fulfillment, and tracking services, to other value-added commerce services, such as rating services.

b. **Complex services** typically involve the assembly and invocation of many pre-existing services, possibly found in diverse enterprises, to complete a multi-step business interaction called a business process.[20]

2.4.5 What is SOA?

Service Oriented Architecture (SOA) represents a popular architectural paradigm for applications, with Web Services as probably the most visible way of achieving an SOA. Web Services implement capabilities that are available to other applications via industry standard network and application interfaces and protocols. SOA advocates an approach in which a software component provides its functionality as a service that can be leveraged by other software components.

Components (or services) represent reusable software building blocks. SOA allows the integration of existing systems, applications and users into a flexible architecture that can easily accommodate changing needs. Integrated design, reuse of existing IT investments and above all, industry standards are the elements needed to create a robust SOA.

2.4.6 Basics of SOA

Traditional distributed computing environments have been tightly coupled in that they do not deal with a changing environment well. For instance, if an application is interacting with another application, how do they handle data types or data encoding if data types in one system change? How are incompatible data-types handled? How are incompatible data-types handled?

The service-oriented architecture (SOA) consists of three roles: requester, provider, and broker.

• **Service Provider**: A service provider allows access to services, creates a description of a service and publishes it to the service broker.

• **Service Requestor**: A service requester is responsible for discovering a service by searching through

the service descriptions given by the service broker. A requester is also responsible for binding to services provided by the service provider.

• **Service Broker**: A service broker hosts a registry of service descriptions. It is responsible for linking a requestor to a service provider.

2.5 Service-Oriented Architecture (SOA)[11]

SOA stands for Service-Oriented Architecture (Sun Microsystems, 2004). It is the architectural style which supports communicating between collections of software services which are available in the network and independent of their implementation and platform. A service is a unit of work done by a service provider to achieve desired end results for a service consumer. The communication of SOA can involve two or more services coordinating some activity or simple data passing. It defines how the two computing entities, such as programs, interact in such a way as to enable one entity to perform a unit of work on behalf of another entity.

Service interactions are defined using a description language. Each interaction is self-contained and loosely coupled, so that it is independent of any other interaction. It is

the style of building reliable distributed systems that deliver functionality as services,

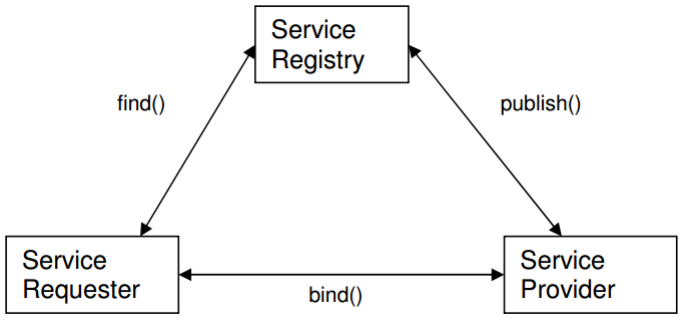
with the additional emphasis on loose coupling between interacting services.

2.5.1 SOA Model

The simple model of SOA contains three entities such as Service Requestor,

Service Registry and Service Provider, and three operations such as find(), bind()

and publish ().



[11]

Figure 3: **SOA model**

The Service Requester is an application that wants to receive a service from another application. It uses the find() operation to find a service description published to one or more service registries by other applications called service provider. The service requester does not need to know where the service is located and how it is implemented, it only gets the service description with the find() operation and uses it to bind to or invoke the services hosted by the service providers.

Any consumer of the services is considered as service requester, it is basically the client side of the communication of SOA.

The Service Registry is responsible for storing the service description published to it by service provider and for allowing service requesters to search the collection of service description contained within it. Based on the search criteria that has been submitted by a service requester in find() operation, it provides the description of the service to the requester. The service description contains the contact information for services. Once the service requestor gets the service description which also contains service connection details, the service registry is no longer needed in the picture, the rest of the interaction is directly occurs between the service requestor and service provider. The service requester uses the connection details to bind() to the service provider. Once bound, they can send messages and receive responses.

The Service Providers creates the service description and publishes it to one or more service registries using the publish() operation. It also receives the service invocation messages from one or more service requestors and sends responds to them after being bound. The service provider works like the server in the client-server architecture.

Besides Web Service, there are many other protocols that can be used to implement the SOA. Such as DCOM (Distributed Component Object Model, which is Microsoft specific), RMI(Remote Method Invocation) and JINI (which are Java specific), and OMG's CORBA (Common Object Request Broker Architecture, which is platform and language independent). CORBA has been successfully applied in various areas ranging from telecommunications, e-commerce, to healthcare etc. However, the biggest challenge faced by CORBA is that it is hard to find a unique RPC middleware to support all the programming languages and all the platforms at a reasonable price [9]. CORBA only supports for UNIX and Windows platform, and Visual Basic does not provide any support for CORBA and therefore limits its usage.

2.5.2 Advantages of SOA

SOA provide several significant benefits for distributed enterprise systems. The most notable benefits of SOA include: interoperability, efficiency, and standardization. We will briefly explore each of these.

i Interoperability

Interoperability is the ability of software on different systems to communicate by sharing data and functionality. SOA/Web Services are as much about interoperability as they are about the Web and Internet scale computing. Most companies will have numerous business partners throughout the life of the company. Instead of writing a new addition to your applications every time you gain a new partner, you can write one interface using Web service technologies like SOAP. So now your partners can dynamically find the services they need using UDDI and bind to them using SOAP. You can also extend the interoperability of your systems by implementing Web services within your corporate intranet. With the addition of Web services to your intranet systems and to your extranet, you can reduce the cost integration, increase communication and increase your customer base.

ii Efficiency

SOA will enable you to reuse your existing applications. Instead of creating totally new applications, you can create them using various combinations of services exposed by your existing applications.

They will not have to spend a lot of time learning every new technology that arises. For a manager this means a reduction in the cost of buying new software and having to hire new developers with new skill sets. This approach will allow developers to meet changing

business requirements and reduce the length of development cycles for projects. Overall, SOA provides for an increase in efficiency by allowing applications to be reused, decreasing the learning curve for developers and speeding up the total development process.

iii Standardization

For something to be a true standard, it must be accepted and used by the majority of the industry. One vendor or small group of vendors must not control the evolution of the technology or specification.

Most if not all of the industry leaders are involved in the development of Web service specifications. Almost all businesses use the Internet and World Wide Web in one form or another. The underlying protocol for the WWW is of course HTTP. The foundation of Web services is built upon HTTP and XML. Although SOA does not mandate a particular implementation framework, interoperability is important and SOAP is one of the few protocols that all good SOA implementations can agree on.

2.6 Web Service

A Web Service is an interoperable unit of application logic that transcends programming languages, operating systems, network communication protocols, and data representation dependencies and issues.

Web Services are based on the following industry standards:

a. eXtensible Markup Language (XML);
b. Simple Object Access Protocol (SOAP);
c. Web Services Description Language (WSDL);
d. Universal Description, Discovery, and Integration (UDDI).

Web service is the specific implementation of the SOA in which service interfaces are stored using Web Services Description Language (WSDL), data is transmitted using Simple Object Access Protocol (SOAP) over Hypertext Transfer Protocol (HTTP or HTTPS), and Universal Description Discovery and Integration (UDDI) is used as the service registry (Lewis &Wrange, 2004).

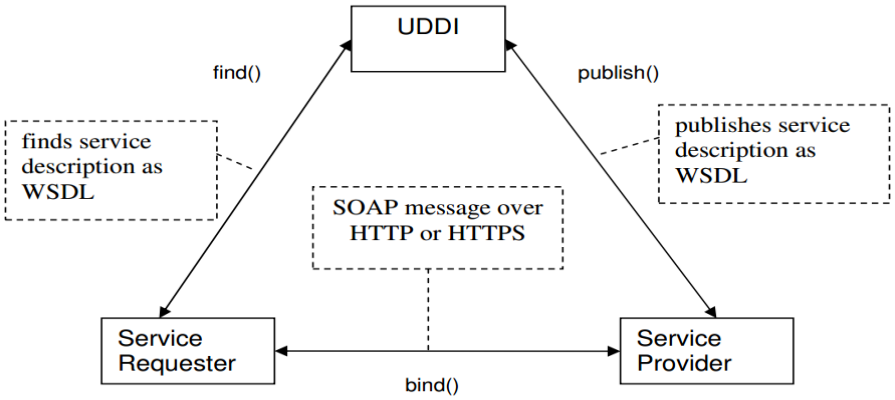


Figure 4: **Web Services – the SOA implementation**

It is an infrastructure for developing and deploying distributed applications. Web services are typically intended for applications consumption, in contrast with contemporary web applications which are meant for human users (Loughran, Gudivada & Kalavala, 2005). The implementation of SOA using web services is shown in Figure 2 (Barry, 2007).

2.6.1 Components of Web Services [12]

Web services are built on foundation of different, but cooperating specifications and standards. web services are comprised of the specifications and standards of HTTP, XML, SOAP, WSDL, UDDI etc. in its definition. They are the key components of web services and the brief descriptions of these components are as follows:

a. Hyper Text Transfer Protocol (HTTP)

is the workhorse of the web. The purpose of HTTP is to provide a protocol to move requests and responses between web clients and servers. It carries any information that is placed in it without regard for its data type. As a result, it is popular way to transport SOAP messages between client and web services.

b. eXtensible Markup Language (XML)

XML has become the common language of the computing environment. An XML document is well-formed if it conforms to the basic syntax rules of XML, and is valid if it is well-formed and also conforms to the rules defined in the XML Schema. Parsers are needed to work with XML document validation, and act as an interface for traversing and accessing the document.

c. Simple Object Access Protocol (SOAP)

SOAP is a message transmission protocol that enables method calls to be sent in an XML format from one computer to another in a decentralized, networked environment and permits a communication in heterogeneous universe. It also can send an entire XML document or data instead of a method calls. It is protocol which it is light, simple, easy to deploy, extensible and open.

A SOAP message travels between SOAP nodes on a SOAP message path from an initial sender through one or more intermediate nodes to an ultimate receiver. The message body is processed by the ultimate receiver.

d. Web Service Description Language (WSDL)

WSDL is a specification that describes web service information such as what methods are available, what parameters they take, where it resides and how to invoke it. A WSDL document is an XML document that contains all the information that the service requestor need to contact and invoke a service. In addition, it is a platform and language neutral.

WSDL separates the description of the abstract functionality offered by the service from the concrete details like how and where this functionality is offered. WSDL is extensible and easy to consult over the Internet. A programmer or program is able to read this document and create an unambiguous message that can call a method or methods in this service.

e. The Universal Description, Discovery, and Integration (UDDI)

UDDI is a XML specification. It describes a special type of registry that lists web services in which the service requestor might potentially be interested. UDDI is a catalogue that contains the services offered by the enterprises over their web sites. This catalogue can contain quite a bit of information such as description, specification (contract), classification, usage history, test results, performance metrics, documentation etc.

This information allows the web services to be searchable based on various criteria.

It uses special classification schemes called taxonomies that categorize a web service in ways that are meaningful to potential clients.

2.6.2 Security Concerns

If an organization uses the advantages of web services, it must trust the security of the web services. To use the web services for its information systems needs, a firm must provide access to its information assets. This action can become an attractive target for malicious

hackers, industrial espionage and fraud. The assurance of security of web services is necessary for the organization to be willing to adopt a security technology.

Web Services Security Requirements

The requirements for information security (i.e. Confidentiality, Authorisation and Authentication, Integrity) remain the same for web services. To ensure persistent security, SOAP messages must include information about the message's security requirements.

i Web Services Security Technology

SOAP technology is built on XML, a reasonable approach to Web Services would assure the information security requirements for an XML message. This section presents an overview of the information security technologies available to assure the security of an XML message.

ii Confidentiality for Web Services

XML encryption, a specification produced by the World Wide Web Consortium (W3C), is used to encrypt portions of XML documents. XML encryption assures confidentiality in the case of any security context beyond a simple HTTP/SSL connection.

iii Integrity for Web Services

XML signature, a specification produced jointly by W3C and the Internet Engineering Task Force (IETF). An XML signature is the XML equivalent of a digital signature. It is used to digitally sign selected portions of an XML document thereby ensuring integrity.

iv Authentication and Authorization Web Services

A web services user can request services from any number of different service providers. That user should be authenticated on each service provider's system. This authentication can then be used to determine the resources that a user is authorized to access on a particular service provider's system.

2.6.3 How Web Services Work [12]

The Web-services framework is divided into three areas – communication protocols, service descriptions and service discovery, of which each is specified by an open standard [26].

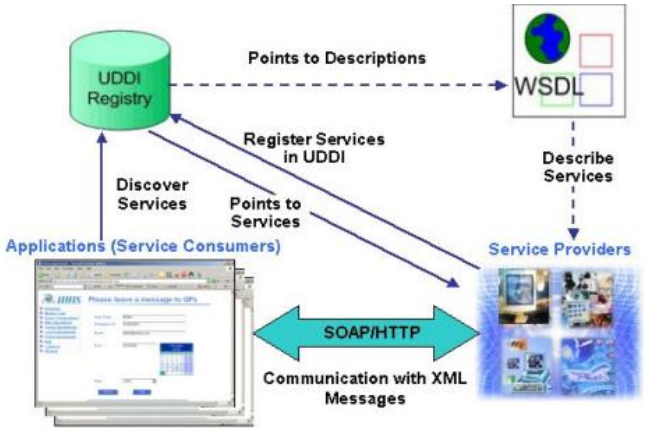


Figure 5: **Web Service Technologies**

Figure 1 In general, web services consist of two major technologies (XML – eXtensible Markup Language and SOAP –Simple Object Access Protocol) and two assistant technologies (WSDL – Web Services Description Language and UDDI – Universal Description, Discovery and Integration).

* Firstly, service providers would make use of WSDL to describe their web services.

Following the above step, service providers would register and publish their services in UDDI

* Applications or service consumers find services via UDDI which would direct service consumers to relevant services according to the description of web services

* Regarding to previous step, applications or service consumers are able to invoke relevant web services using SOAP transmitted via HTTP on the Internet. Web services encoded in XML, SOAP provides a way to communicate between applications developed with different programming languages and running on different operating systems. In fact, Web services provide a distributed computing technology for integrating applications on the Internet using open standards and XML encoding. The use of standard XML protocols makes Web services platform-,Language- and vendor-independent, thus an ideal solution for use in application integration.

Figure 6 depicts how applications work with Web Services.

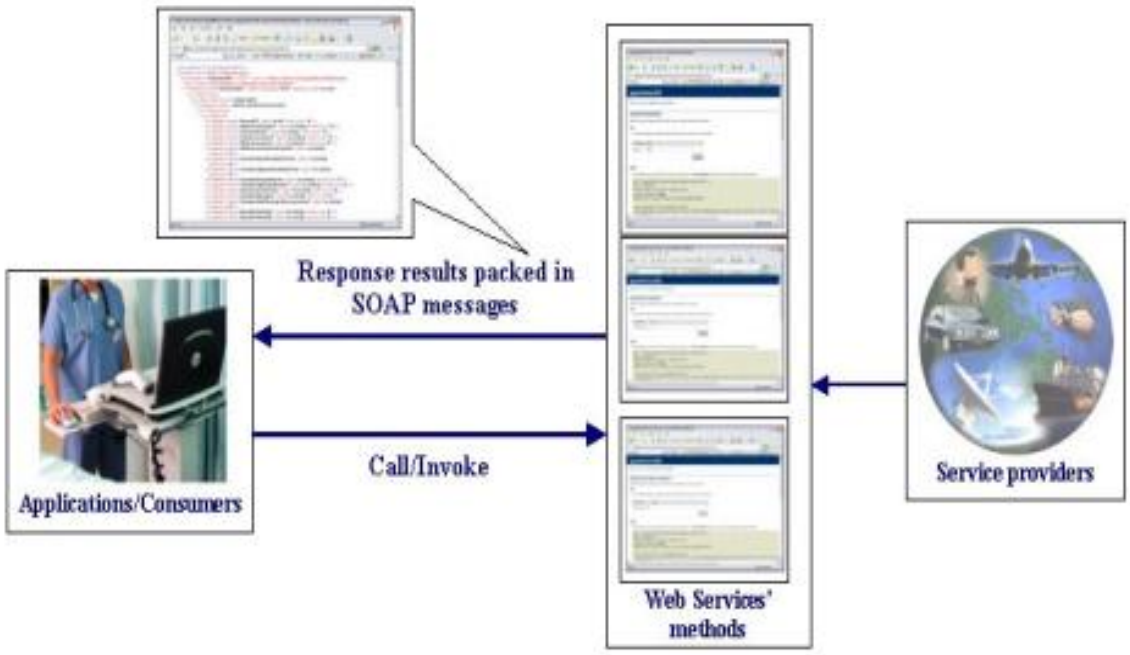


Figure 6: **Interaction between applications/consumers and Web services.**

With respect to Fig.5, applications send requests and responses to and from Web services via SOAP.

When a program invokes a Web-service method, the request and all relevant information are packed in a SOAP message and sent to the appropriate destination.

When the Web service receives the SOAP message, it begins to process the contents (called the SOAP envelop), which specifies the method that the client wishes to execute and the arguments the client is passing to that method. After the Web service receives this request and parses it, the proper method is called with the specified arguments (if there are any) and the response is sent back to the client in another SOAP message. The client parses the response to retrieve the result of the method call, Service provider and the Service registry occur in SOAP over HTTP or HTTPS using the WSDL interface and this particular instance of SOA is considered as web service.

Web services define a set of specifications that provide an open XML-based platform for the description, discovery, and interoperability of distributed, heterogeneous applications as services. It provides information to applications rather than to humans, through an application-oriented interface. As the information is structured in XML, it can be parsed and processed easily by the application.

2.7 Evolution of Enterprise Application Integration Architectures

Application Integration Architectures are middleware software topologies that help different systems communicate with each other. There are three topologies for integrating applications we will have a look at and their advantages and disadvantages. Point-to-point topology which was firstly used and seen as the most basic solution, hub-and-spoke topology administered from a system that is center of integrations, bus topology which is the architecture of enterprise service bus.

2.7.1 Point to Point Topology

Most of integration projects are the results of the need for communication between two systems. The most practical way of providing this communication is to utilize Point-to-Point Connection. In point-to-point connection only one receiver get one particular message providing that the system knows where that particular message to be delivered. The sending system usually has to transform the message into the format which could be understood by the receiving point.

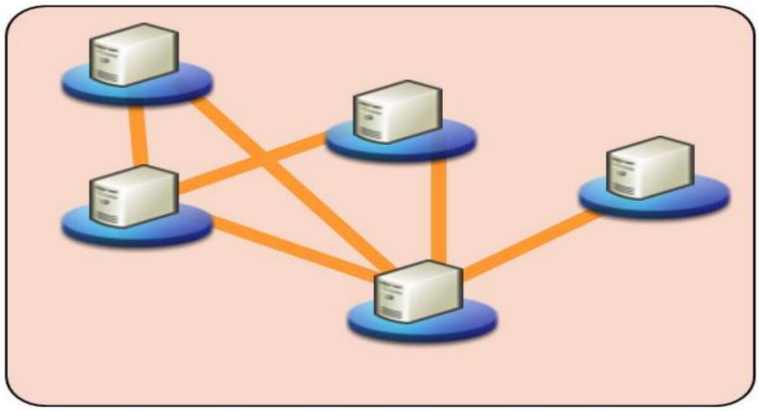


Figure 7: **Point To Point Topology**

In point-to-point connections, the addresses of all nodes or points that need to be linked are determined by the system. If there are changes in target addresses or protocol details, an update is required for the systems. Furthermore, if the integration network grows larger and at the same time changes become recurrent, it is likely that operational cost of maintaining system adopting this approach becomes notable.

In most of the integration projects, data is expected to be transformed between the source system and the target system. Moreover, developers sometimes may want to make use of some conditional logic while customizing message routing. In point-to-point connections, a

duplication of aforementioned logic is present on each server in need of transformation and routing yet writing a duplication code might be costly, hard to maintain and to test [23].

Advantages

a. Integration is the simplest of all and tightly bound
b. Enables better integration with small number of systems.

Disadvantages

a. There is limited flexibility and constant need for updates.
b. The more integration points to take care of the more complex it gets.

2.7.2 Hub and Spoke Topology

In Hub-and-Spoke topology, there is a centralized broker which is called a hub and there are adapters, namely, spokes which enable applications to connect the Hub and they convert the formats of the application data to that of the Hub recognizes, or vice versa. The Hub deals with all messages, their transformation processes into the format that destination application understands and the routing. Spokes get data from the origin application as relay messages to the Hub, then the Hub passes those messages to a subscribing adapter and it send those over to the target application.

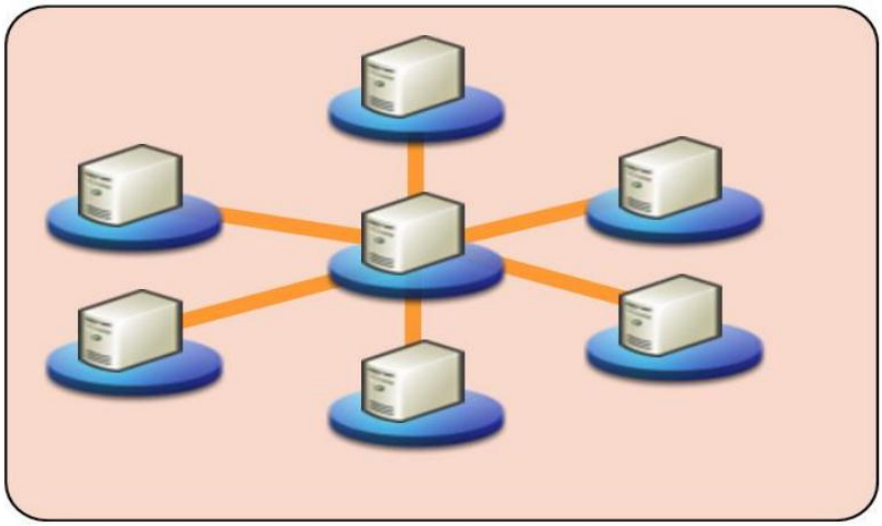


Figure 8: **Hub and Spoke Topology**

To create a central location for control, hub topology is very helpful and the source sends the messages to the central hub. Hub topology is very effective provided that enterprise events are not dependent and if a single vendor provides the Message Oriented Middleware (MOM). The source application here forms a message in a particular format and the hub re-forms and sends it to the spokes linked to the hub [24].

Advantages

a. Enables integrations via central management.
b. Less complexity compared to point to point.
c. Business process is controlled and mapping in data layer is provided
d. There is more scalability.

Disadvantages

a. All system is susceptible to single point of failure.
b. There is limited scalability for technologic infrastructure
c. The available hubs cannot generally deal with the incoming transaction duties from other sources except the middleware they work on.
d. Integration processes with multiple sources and destinations are hard to manage.
e. In need of a database, processing or routing bottlenecks crowd the hub since volumes grow and integration rules get more complex.

2.7.3 Bus Topology

Messages from source applications are put onto a system-wide logical software bus that other applications can access. That's why, bus topology is beneficial for relaying information to multiple destinations. Messages on the bus can be particularly subscribed by multiple applications and the data relay may not have to pass through the central switching point, which is possible only in publish and subscribe middleware.

The glitch of bottlenecks is, however, overcome by bus topology.

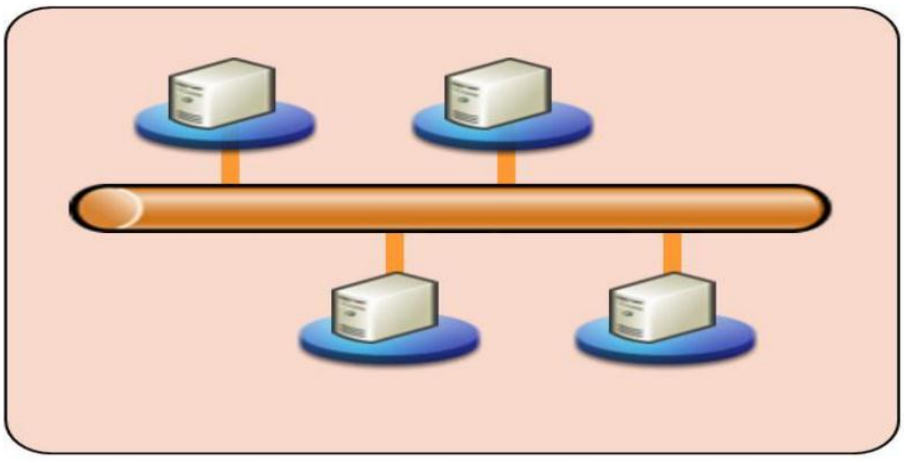


Figure 9: **Bus Topology**

A central messaging bus is utilized for the distribution of messages by bus architecture and the messages are published by applications to the bus using adapters.

The message bus takes these messages to the subscribing applications which contain adapters taking the messages and re-forming them into the required format [25].

Advantages

a. Enables integration of loosely coupled services.
b. Enables infrastructure for shared communication
c. Service Meditation

Disadvantages

a. It is hard to control all messages on bus
b. It is hard to adapt systems to loosely coupled services
c. Latency period is increased compared to point-to-point integrations

Before touching upon the definition of Enterprise Service Bus (ESB), it is essential to clarify what Service Oriented Architecture (SOA) is and what features it provides since in order to elaborate more, it is useful to define Enterprise Service Bus first.

2.8 Enterprise Service Bus

"An Enterprise Service Bus (ESB) is software infrastructure that enables SOA by acting as an intermediary layer of middleware through which a set of reusable business services are made widely available."[22] Mike Gilpin, Forrester Research, August 2004

Enterprise Service Bus (ESB) is a platform that gathers messaging, web services, data transformation and intelligent routing in a way that links numerous different applications across an organization and its partners and coordinate them while keeping transactional integrity. It makes use of the features provided by Service Oriented Architecture (SOA), Enterprise Application Integration (EAI), Business-to-Business (B2B), and web services, thus making itself an integrated platform enabling essential interaction and communication services that complicated software applications need via an event driven and standards-based messaging engine, or bus, built with middleware infrastructure product technologies. By insulating the link established among a service and a transport medium, it is utilized to realize the needs of service-oriented architecture (SOA) [26].

ESB is the message-based integration architecture containing SOA features and supporting its infrastructure:

i. Agility
ii. Flexibility
iii. Reusability

Basic features of SOA, form the foundation of the advantages provided by the ESB architecture. These three are the features that are targeted and benefitted as much as possible by the infrastructure that is the focal point in my thesis. SOA can also be seen as an architectural format that backs weakly coupled services in providing flexibility in businesses in a way that enables interoperability in an multiple-technology environment. SOA is comprised of a complex group of business-aligned services enabling the actualization of adaptable and customizable business processes by utilizing interface-based service descriptions [27].

The aim of getting SOA to deepen IS and business activity, and to ameliorate IT-business alignment in multi-atmosphere business conditions is not very explicit in the definition. SOA differs from other ITs in that it accentuates more on IS agility thus ameliorating business agility. The closer the link between IT and business, the more quickly an organization can act to alter IS applications according to business needs.

SOA provides methods for systems development and integration where systems group functionality around business processes and package these as an interoperable service [28]. An organization can make use of these services by re-using them or these might also be commercially on the market. Thus, SOA separates functions into distinct units, or services, which developers make accessible over a network in order that users can combine and reuse them in the production of business applications. [29]Between these services, there is always a strong communication which consists of data exchange, and enables coordination of an activity processed in two or more services.

Data transfer between reusable services and cross domains is easily achieved. Reusable services reduce integration costs in SOA architecture and facilitate the integration of end points to the system. Many end points in SOA architecture contain single service availability for use. That's why each implemented service is designed and developed independent of business flows, other enterprises or technologies. Services which are on SOA architecture and can be called from more than one place, by increasing the reusability, become available for the use of multiple external systems at the same time, and enable updating of the changes on the whole system with a single move when there is a need for change.

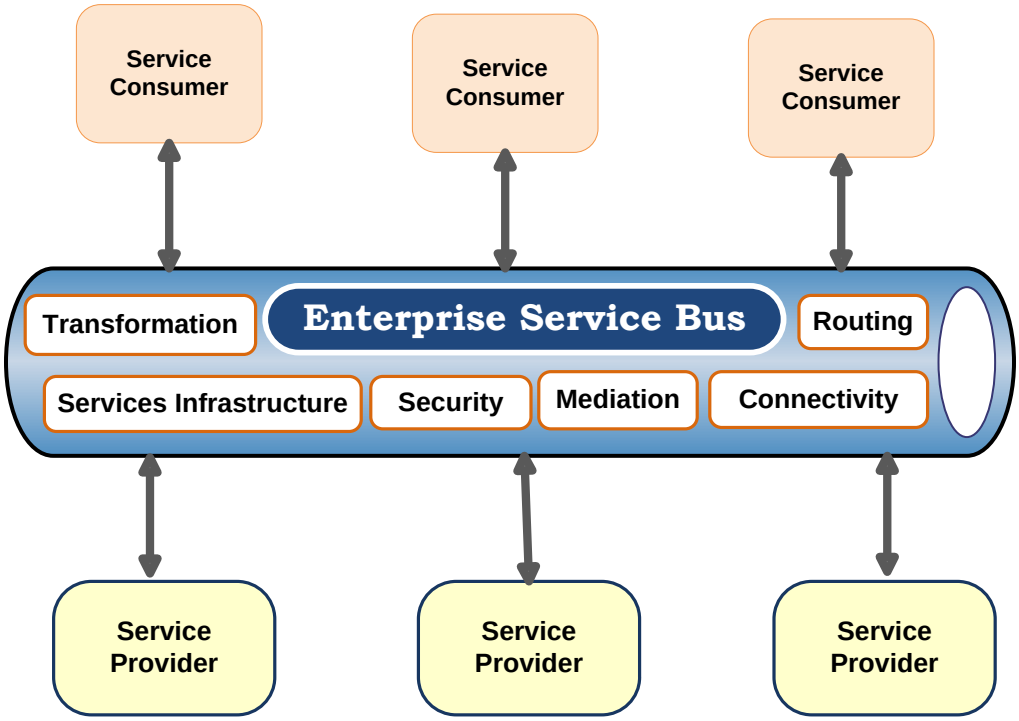


Figure 10: **Enterprise Service Bus Architecture**[30]

In our research we see an ESB as an integration infrastructure which is used to facilitate SOA. An ESB combines service hosting, message transformation, protocol bridging and routing to connect and coordinate the interaction of a significant number of diverse applications across extended enterprises [31].

ESB greatly enhances the usage of SOA with a virtual bus to connect many disparate systems and services together (Chappell, 2004; Schmidt et al., 2005; Menge, 2007; Bygstad and Aanby, 2010). ESB is message-based bus architecture which consists of a set of software components called service containers (Menge, 2007; Bygstad and Aanby, 2010). Service containers are interconnected over a reliable and secure messaging channel. A service container connects one or many software services or systems through service adapter(s) (Menge, 2007). A system or service sends request messages directly to its connected service container which inturn processes and routes the messages to a destination (requested) service container. The destination service container processes and forwards the messages to its connected destination (requested) service or system through a service adapter[31].

A service container contains several modules namely: connectivity adapter module, mediation module, message routing module, security module, and management module. Each module is responsible for specific tasks.

I. **Routing** – its routes a request to particular service provider based on a static or variable routing criteria example of types of routines:

a. Static routing
b. Content Based Routing
c. Complex Rules based routing

Determines appropriate end consumer based on preconfigured rules or dynamic created request.

II. **Message Transformation**– converts the structure and the format of the incoming service request to the structure and format of the outgoing message that will be consumed by the service provider. Data transformation between canonical data formats and specific data formats required by each ESB connector. An example of this would be transforming between CSV, Cobol copybook or EDI formats to either SOAP/XML or JSON. Canonical data formats greatly simplify the transformation requirements associated with a large ESB implementation where there are many consumers and providers, each with their own data formats and definitions.

Examples

a. text >> xml
b. text >>json
c. edr>> xml
d. Db table record >>json/xml

III. **Security** – provides protection to enterprise services from unauthorized access by implementing security standards and policies including:

a. Authentication
b. Authorization
c. Encryption
d. Auditing
e. Intrusion detection

This module includes adapters for connecting services that are provided by the ESB via the Service Hosting module. The ESB can be assumed to provide a set of adapters from the ground up, and also has the option for customers or third-party developers to develop additional adapters for customer-specific requirements.

IV. **Connectivity/Transportation** – Services or Systems connect to each other via **endpoints** - uniform, unique "addresses". Messages dispatched between endpoints are using unified **transport** (method/protocol that encapsulates message's payload). Applications that natively use different transport, need to connect to ESB via suitable adapter - service that will provide necessary transport conversion. Transport protocol conversion. ESB seamlessly integrate applications that use different transport protocols (JMS, HTTP/S, pure TCP, etc.)

V. **Mediation** – also refers to data mapping, maps data elements from the source data system to the destination data system usually before transformation occurring. This module contains individual processing units that perform a specific function, such as validating, filtering of messages etc. [31].

2.9 Overview of Web Services and their applications

The term Web Services describes a standardized way of integrating Web based applications using XML, SOAP, WSDL and UDDI open standards over the Internet protocol backbone. Under the web services, a program could viewed as the building block for a more complex program and this continues in a recursive fashion. It indicates a distributed technology, which entails bridging across different technologies, platforms and enterprises. They are neutral with respect to object models and types, and provide synchronous communication of high-level, semantically rich, XML business documents.

Web services has today diversified into various e-business (for instance e-bay API, Amazon API), searching tools (Google API) and even programming assignment cheat checker developed by UC, Berkley).

One can search http://www.uddi.org or http://www.methods.com for web services currently implemented by various individual/companies, sorted categorically. Other simple web services that one could easily find on the web include web service return the current time from the UTC and web services to validate US Bank routine number.

The disparate hospital information systems need to be interconnected in order to promote sharing of the required data, web services have been used to provide a remarkable solution.

Below are examples of systems that use web services to enhance interoperability between the unrelated systems.

2.9.1 A SOA based architecture to promote ubiquity and interoperability among Health Information Systems in Portuguese (2011)

The Portuguese healthcare system is divided into two main areas, primary care, represented by health centers, and differentiated care, represented by hospitals and other healthcare institutions. Primary care is the primary source of care and the patients should consult them first, in general care and proximity with the community, routine consults, monitoring and follow up of chronic patients, along the patient's life.[14]

The access to differentiated care is usually performed by the intervention of primary care health professionals or in case of emergency. This type of care is more specialized than those in primary care and they are used when the primary care can't give an adequate response in specific situations like surgery or oncology patients.

The healthcare professional in a hospital is unable to access the information created in a health center. As a rule a patient starts to take medication by indication of the primary care physician, but when the patient needs to go to a hospital, for whatever reason, the hospital's health professional doesn't know if the patient is on any medication and what is the medication, unless the patient informs him.

The medical information created in a hospital is not accessible from a health center and vice-versa.

To promote interoperability among heterogeneous health information systems and the inclusion of mobile devices in health care, two approaches have been followed. a procedural approach and a documental approach. The procedural approach uses web services to promote the integration of applications using specifications such as Simple Object Access Protocol (SOAP), Web services Definition Language (WSDL) and Universal Description, Discovery and Integration (UDDI). The documental approach tries to describe in detail the elements of the exchange of information among the different systems.(Mykkanen et al, 2007).

The Total Health Enriching Mobile U-health Service System (THE-MUSS) is a system for ubiquitous solutions to the area of healthcare. The system consists of a Business Process Management System (BPMS), mobile devices, biosensors and a set of primitives defined using web services.

There is also iCabiNET system that monitors medication intake in outpatients, the system is a mobile application that communicates with a server, where the health information system is running, using web services Representational State Transfer (REST), SOAP or Java

Remote Method Invocation (RMI), the communication is bidirectional, the system warns the patient when is necessary to take a medication and what medication, the application confirms the medication's intake or not the communication takes place via
Bluetooth or the Internet.

The Hospital (H), Primary Care (PC) and Medical Emergency (ME) subsystems represent the hospitals, health centers and medical emergency respectively. The GLOBAL subsystem is where the patient can access his medical and personal information. The subsystems of the different institutions replicate the information in their databases with the GLOBAL subsystem database on a daily basis. The health professional, namely emergency medical, may quickly access all of the patient's information, present in GLOBAL, through a mobile device, using the access interface(web service) for mobile devices (WSM). To update the ME subsystem one must access the services provided by him. The H, CS and ME subsystems already existing were not modified, to promote interoperability between them another piece was created, the BROKER. The BROKER encapsulates the interface and its operations that a subsystem uses to obtain information from another subsystem.

The prototype the architecture provides interoperability in three specific cases, consultations in the CS subsystem, surgery/hospitalization in the H subsystem, and an emergency situation outside a health institution in the EM subsystem.

For example if a patient was submitted to a surgery, the medical information create during that procedure will be stored in de subsystem H and will be accessible to the other subsystems through the operations provided by the BROKER. The same happens with the information stored in the other subsystems.

The focus of the prototype implementation was the BROKER in the H, PC and ME subsystems and in the communication between them as well as the access to the GLOBAL and ME subsystems using a mobile device. To promote interoperability between the different subsystems were used specifications such as WSDL and SOAP.

2.9.2 Web Service-Based Integrated Healthcare Information Systems (WSIHIS) – OPRA Project Queen Marry Hospital, UK (2009)

A scheme called OPRA (OsseointegrationProgramme for Rehabilitation Amputee) was developed at Sweden that consists of patients' selection and recruitment, surgical plan after surgeon retirement and rehabilitation. The whole procedure of Osseointegration from the operation to the rehabilitation of patients lasts years. [13]Overall, this process would involve

doctors, patients, surgeons, prosthetic clinician and rehabilitation clinician. All relevant information must be recorded and documented for progress review. Patients' files are also required if the infection is developed in later stage. They would require an information system to manage and process these massive health data, as the patient, the hospital and the rehabilitation centre and prosthesis are normally not in same location. The data exchange between them would be very important.

Consequently, WSIHIS was implemented by National Program for IT (NpfIT) to computerize all documents and data, and offer a secure and stable environment for the communication between doctors and patients across the Internet. WSIHIS would be required to link with various medical-relevant people (e.g. GP, Surgeon, prosthetic clinician and rehabilitation clinician), who might be using different systems on their desktops to access to WSIHIS for different purposes.

Web service plays as the role of middleware that hides all these differences in system platforms, programming languages and database systems to users and developers. Accordingly, from users' perspective, they get the access to WSIHIS regardless of their different system platforms. From developers' perspective, they can invoke or reuse applications of WSIHIS in their systems with the support of web services.

WSIHIS is built upon Microsoft .Net platform, and Web service plays a role of middleware. Web services can be exposed from and consumed by any platform that can format and parse an XML message because using XML for the formatting of requests and responses. This allows XML-based Web services to bring together disparate pieces of functionality – existing or new, internal or external to an organization – into a coherent whole. Core technologies of Web service are XML and SOAP. Once Web services receive requests from applications, web services would retrieve data from different DBMSs (e.g. SQL server, Microsoft Access, Oracle, etc.) into datasets according to requirements of applications. All datasets would be written in XML messages, in additions, SOAP would act as an XML envelop to wrap those XML-based datasets into SOAP messages. And then these SOAP messages would be transmitted back to applications via HTTP. SOAP in Web service-based middleware provides a way to communicate between applications developed with different programming languages and running on different operating systems.

Additionally, Microsoft .Net platform is another important part of solution for the interoperability in WSIHIS. It is language neutral. It is best thought of as an open

programming platform into which a variety of languages can be plugged. It is achieved by translating all different programming languages into a common language called Intermediary Language (IL).

Major benefits from the achievement of interoperability are firstly, for existing HISs, medical data and functions can be easily shared and exchanged between different HISs, additionally, systems would be accessible to people using different operating systems. Secondly, for those HISs under the expansion and improvement, new technologies can be easily plugged into original systems. Finally, for those HISs under the development, they can reuse functions from other HISs to reduce the development time scale and cost.

2.9.3 Artemis

ARTEMIS is a STREP project supported in the 6th Framework by the European Commission. ARTEMIS aims to develop a semantic Web Services based interoperability framework for the healthcare domain. Artemis addresses the interoperability problem in healthcare domain in two respects: [15]

First, in Artemis infrastructure healthcare institutes keep their proprietary systems and expose their medical applications as Web services. Web services provide functional interoperability through well accepted standards like SOAP and WSDL.

Secondly, Artemis provides the interoperability at the semantic level through semantic annotation of service messages and functionalities through OWL-S and ontology mediation.

Artemis has a peer-to-peer architecture in order to facilitate the discovery of healthcare web services. In Artemis, healthcare institutes are represented as peers.

Artemis peers provide interfaces to the healthcare information systems to enable them to discover and consume the Web services provided by the other peers.

In order to facilitate the discovery of the Web services, there is a need for semantics to describe what the service does, in other words what the service functionality semantics is in the domain. For example, in the healthcare domain, when a user is looking for a service to admit a patient to a hospital, he should be able to locate such a service through its meaning, independent of what the service is called and in which language.

In Artemis, HL7 categorization of healthcare events are used to annotate Web service functionality since HL7 exposes the business logic in the healthcare domain. OWL-S Release 1.1 also indicates that service characterization must effectively position a

service within the broad array of services that exists within some domain, or perhaps in the world at large10. OWL-S proposes to construction of a "Service Profile hierarchy", with inheritance of properties by subclasses as a technique for this kind of service characterization. In the same manner, we have created the HL7 event based Artemis Functionality Ontology as a "Profile Hiearchy". If further ontologies are developed for this purpose, they can easily be accommodated in the Artemis architecture through ontology mapping.

When invoking a Web service, there is also a need to know the meaning associated with the messages or documents exchanged through the Web service. In other words, service functionality semantics may suffice only when all the Web services use the same message standards. For example, a "GetClinicalInformation" Web service may include the messages to pass information on diagnosis, allergies, encounters and observation results about a patient. Unless both the sending and the receiving ends of the message conform to the same EHR standard, interoperability can not be achieved.

For this purpose in Artemis, the input and output parameters of the Web services defined in OWL-S are annotated through message ontologies. In Artemis Message Exchange Framework (AMEF) 2, the messages which may be in EDI or XML are normalized to messages represented by the messages in OWL. The most powerful aspect of AMEF is that the healthcare organizations are not expected to conform to a single commonly agreed messaging format. Artemis Architecture provides an OWL mapping tool, called OWLmt11, to handle ontology mediation by mapping the OWL ontologies in different structures and with an overlapping content one into other. In Artemis architecture, healthcare institutes define the mapping between their own message ontology and one of the "Clinical Concept Ontologies" available in Artemis P2P Network. "Clinical Concept Ontologies" are designed based on the prominent EHR based healthcare standards such as HL7 CDA, CEN ENV 13606 1.

Such mappings are defined graphically between source and target ontologies and the mapping definitions produced are advertised to the Mediators in the P2P network. When a peer wishes to invoke a web service, the mediator hosting the OWL mapping engine mediates the web service input and output parameters from the source ontology instances to target ontology instances automatically using the previously stored mapping definitions.

3.0 CHAPTER THREE: METHODOLOGY

3.1 Introduction

In order to accomplish the objectives specified in chapter one, a number of activities were carried out as outlined below:

i. Literature review on health information systems and Web services technology

ii. Gather user requirements.

iii. Analyze business components to be reused or create new service.

iv. Design the Web Service (WS).

v. Develop WS by implementing business logic with the used of interface and implementation classes.

vi. Build WS by wrapping component into WS.

vii. Deploy WS to the target web server based on the deployment script .

viii. Test and debug WS using web service client

ix. Publish WS if publishing to service registry is required.

The development is depicted in the diagram below.

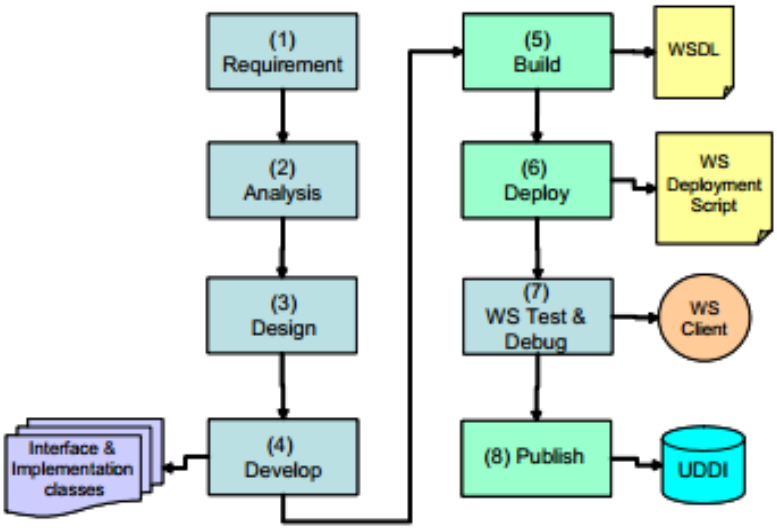


Figure 11: **Web Services Development Workflows**

i. Literature review on health information systems and Web services technology

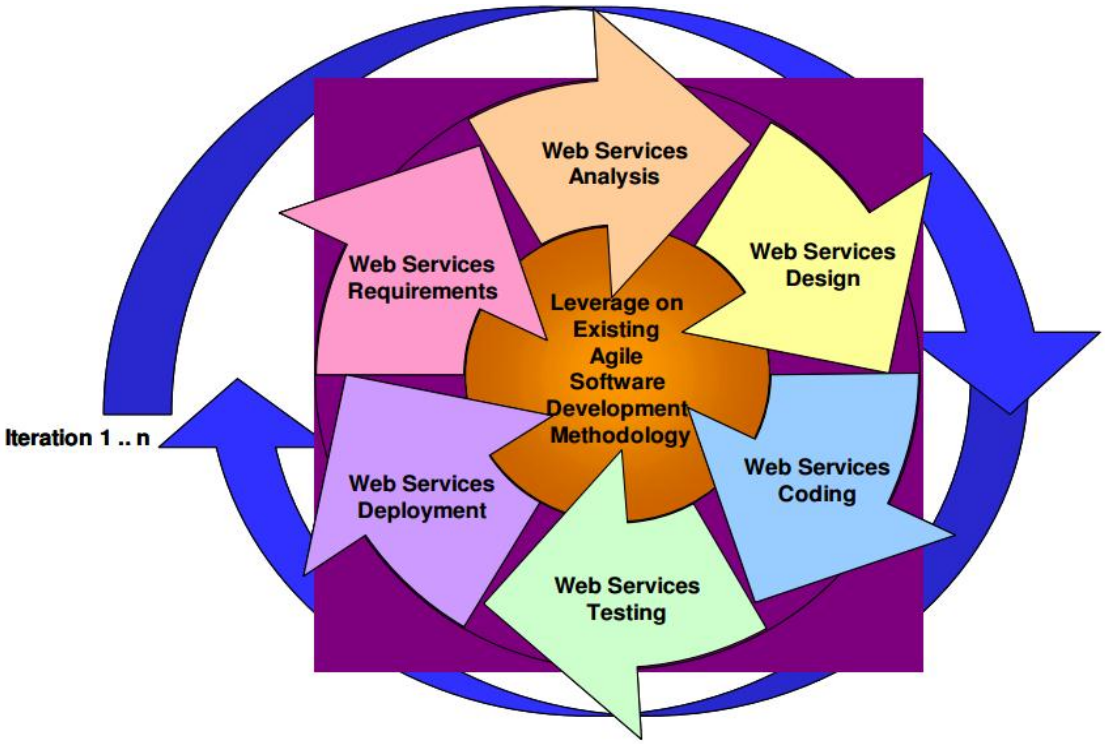
A brief literature review of the health information systems was carried out with a view to gain thorough understanding of the underlying needs, challenges and desired operations level of health information interoperability systems. This review guided the process of

compiling the requirements document. The study identifies the use of Web services development methodology to specify, analyze, design and implement the system.

3.2 Web services development methodology

Web service development methodology defines a set of common practices that create a method-independent framework, which can be applied for developing Web Service applications. The realization of SOA is centered on Web Services (WS), the SOA application development involves developing software components (services) for software reuse and wrapping software components as Web Services for end user applications or other services consumptions. The WS methodology is a detailed process for specifying, designing, and implementing services oriented systems.

According to analysis done by (**S. Poh Lee *et al,* 2007**) on gaps of existing software development methodologies (e.g. they do not include the design and development factors specific for Web Services etc) and study of Web Services characteristics, forms our basis of extending the existing agile software methodology with Web Services. The transition through the development phases will be iterative and incremental in nature to accommodate revisions in situations where the scope cannot be completely defined up front.



[16]

***Figure 12:*Web Service Development Methodology (extending Agile Methodology.)**

Although the phases are described in a sequential fashion it is acknowledged that like most Software Engineering methodologies, practice involves revisiting earlier phases as one works out the details.

Bottom-up development approach will be used as it involves creating a Web service interface from the application programming interface (API) of the application that implements the Web service. Using this approach a new service interface is developed for existing application(s). Bottom is well suited for an environment that includes several heterogeneous technologies and platforms or uses rapidly evolving technologies. An overview of the methodology, including its phases, deliverables, and intermediate products, is depicted below.

The following is description of the phases and deliverables.

A. Requirements Phase consists of the following activities:

- To understand the organizations requirements/needs and translating them into WS requirement in terms of the features, the functional and non-functional requirements, and the WS constraint.
- Identifying Web Services, categorizing the needs into Web Services and the features
- required for the respective Web Services.
- Define use case models for the respective Web Services.

B. Analysis Phases consists of the following activities:

- To refine the requirements further and translate the requirements into conceptual models.
- Architecting analysis is done to define high-level structure and identify the Web Services interfaces contracts.
- Analyzing the granularity of Web Services interface contracts.
- Selecting technology platform for implementation framework.
- Defining Web Services candidate architecture.
- Identify architectural components to be exposed as WSs and specify major information exchanged with client.

C. Design Phase consists of the following activities:

- To detail design of Web Service and refine further the WS interface.
- The interaction of between the service and the client, e.g. asynchronous/synchronous or rpc/document is considered.

. **D. Implementation Phase** consists of the following activities:

- Do the actual coding of Web Services. The wrapping of components APIs to Web Services interface is done.
- The generations of UDDI, WSDL and WS test client are produced. The WS will be deployed to the target application server.

E. Testing Phase consists of the following activities:

- Conduct a complete test for Web Services including functional and non-functional requirements.
- Testing the coded services and processes for functional correctness and completeness as well as for interoperability

F. Deployment Phase consists of the following activities:

- To ensure Web Service is properly deployed to the targeted application server.
- To validate proper deployment of WS, the server specific WS client is used to conduct the deployment.
- Specify if the publishing of their web services is required for internal organization, or extended to their trading partners or external used.
- This leads to the decision whether to have a private or public service registry to serve their company's needs. If there is a need to publish to a service registry, then activity for gathering additional information for registry publishing is considered for the phase
- Execution and monitoring of Web services. This phase includes the actual binding and run-time invocation of the deployed services as well as managing and monitoring their lifecycle.

4.0 CHAPTER FOUR: SYSTEM ANALYSIS, DESIGN & IMPLEMENTATION

In this chapter, we shall describe the system prototype in detail, the processes and mechanisms employed in the prototype in order to achieve the functionalities. We then, describe the bus use case in detail using flowcharts, use case diagrams and activity diagrams. We have relied heavily on open source tools for the development of prototype.

4.1 Software Tool used for development

Operating System

- Linux Platform e.g. Kubuntu (kernel version 2.6++)

Relational Database Management

- MySQL (version 5++)

Programming Language

- PHP (version 5++)

Running and Testing Environment

- Standard Web browser e.g. Mozilla Firefox or any browser

Web server

- Apache

4.1.1 Diagrammatic representation of the system

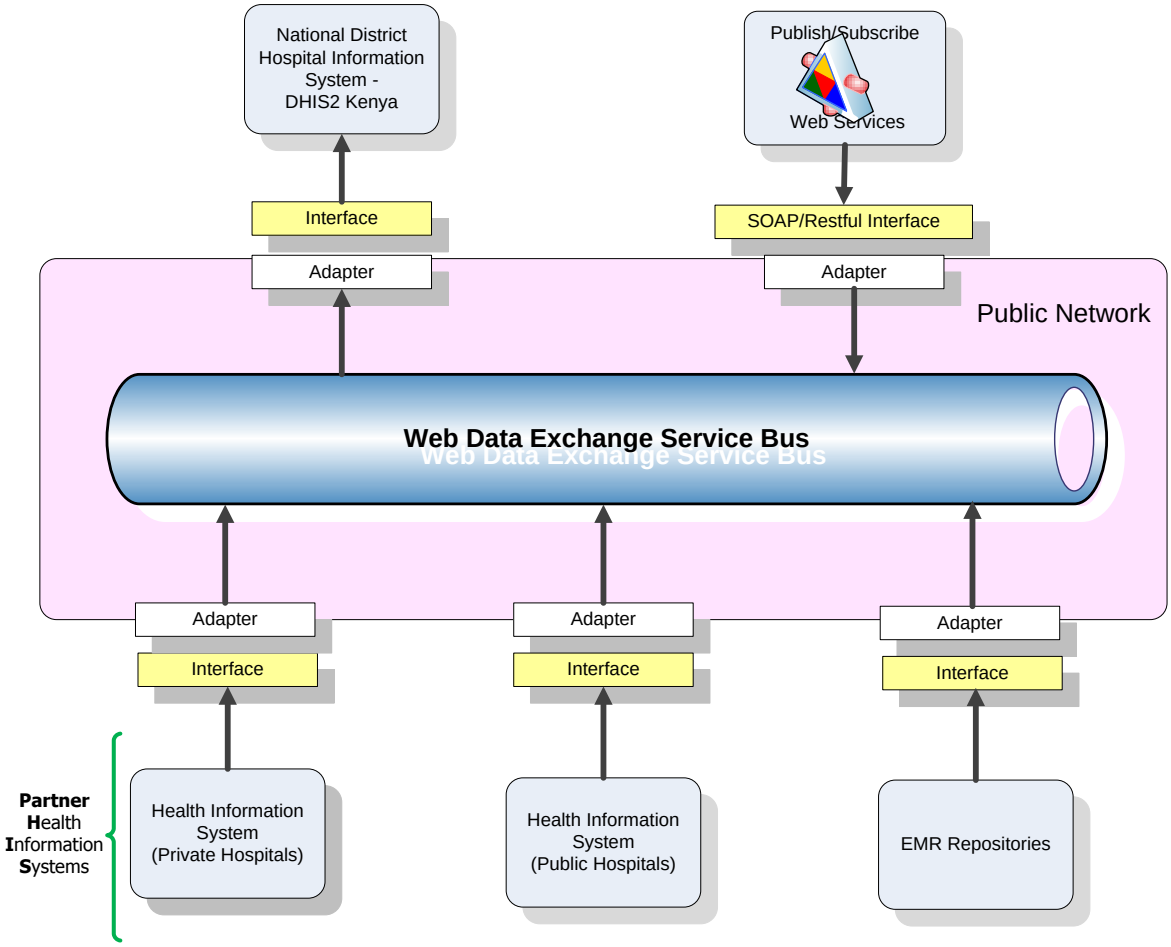


***Figure13*:Diagram of the Web Data Exchange Service Bus for HIS**

4.1.2 User requirements

User requirements specify what the user wants to achieve from the proposed system. These are the problems which the proposed system should solve. The following are the user requirements

i. Provision of secure access to the system
ii. The prototype will remotely login and gain secure access to the National Health Reporting System – DHIS
iii. It will enable the user to view and specify the required parameters available in DHIS for submitting information of a health center.
iv. Name of the health center e.g. St. Mulumba Hospital in Kiambu County Thika district.

v. Period which the health information was collected e.g. month of January

vi. Data set or reporting tool to be used e.g. MOH 711, MOH 730

vii. Allows user to specify way of sourcing reporting information i.e. provide parameters to connect remotely to Partner Health Information: Host Internet Protocol Address or hostname, database name, SQL statement querying the database for the information.

viii. User will source data to the prototype from an excel sheet file by browsing for the file received from a health center.

ix. The user will harvest or pull information from a remote host and initially map the data values to the data elements sourced in a real-time way from the DHIS system.

x. The system will allow the user to send the reporting information from a partner HIS to the National Health System – DHIS.

4.1.3 Use case diagram

The proposed Web Data Exchange Bus for enhancing interoperability between disparate health information systems will enable the users login into bus and specify parameters of the health facility and information to be reported. The system entails a number of use cases which include County Health Record Information Officers (CHRIOs), Web Data Exchange Service Bus (WDESB) system and DHIS2 system. The main users of the system are CHRIOs who receive health aggregated data/summary reports every month from the health facilities located in a county. The summary reports are in form of manually filled data sheets e.g. MOH 710, MOH 711, MOH 730 etc. The data on the summary reports are keyed in into the national DHIS2 system.

The proposed system will be used to extract data directly from heterogeneous information systems used by the health facilities or receive the data in form of a excel worksheet. The system mediates between consumption of the Web Services provided national instance of DHIS2 and the partner Health Information System(HIS).

The use case diagram in figure below shows the interaction of the system and its stakeholders.

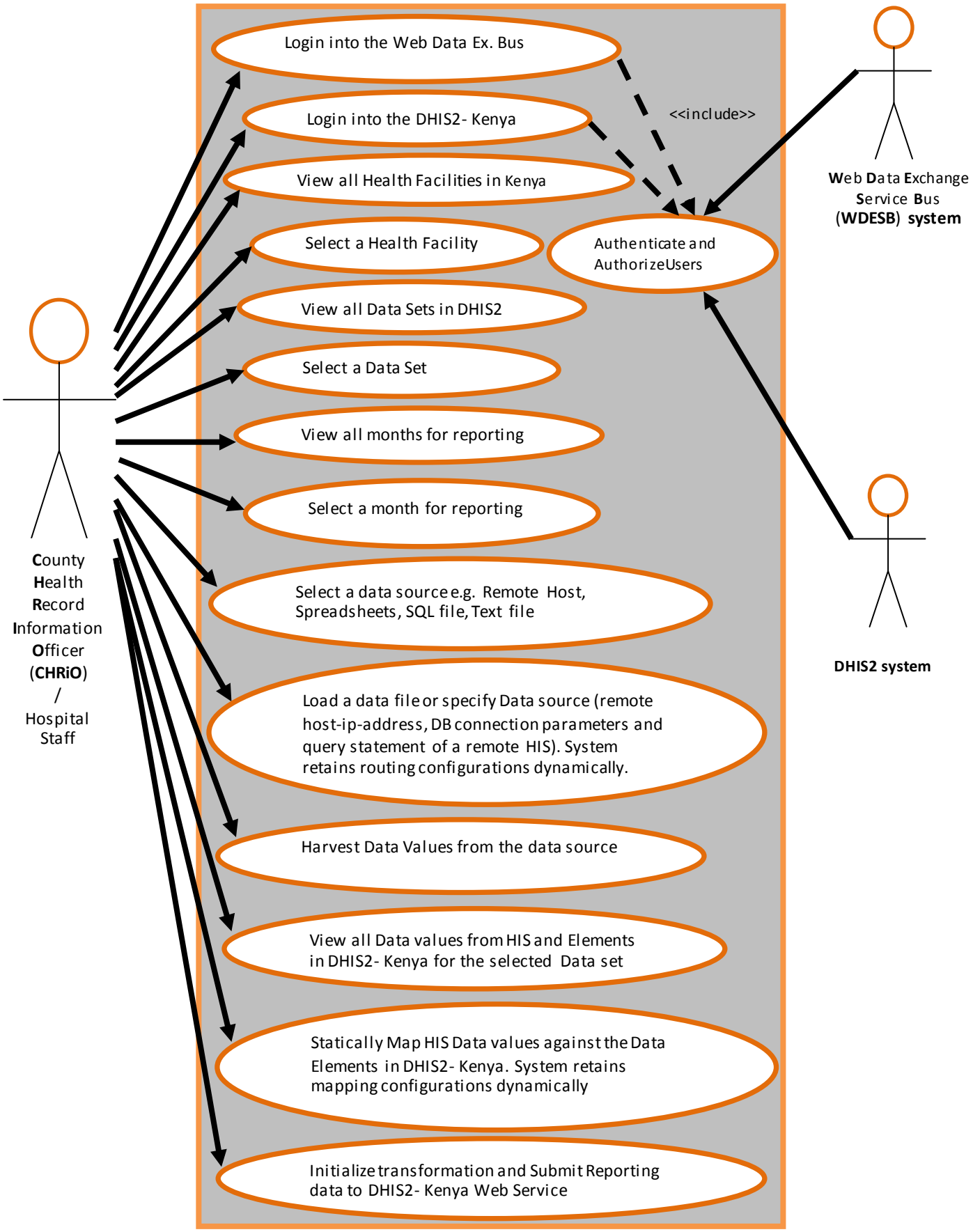


Figure 14: Use Case of the Web Data Exchange Service prototype

4.1.4 The Integration Approach

A multi-tier architecture was employed in devising the solution. The architecture consists of the database layer, health information systems layer, middleware/ bus, and presentation layer.

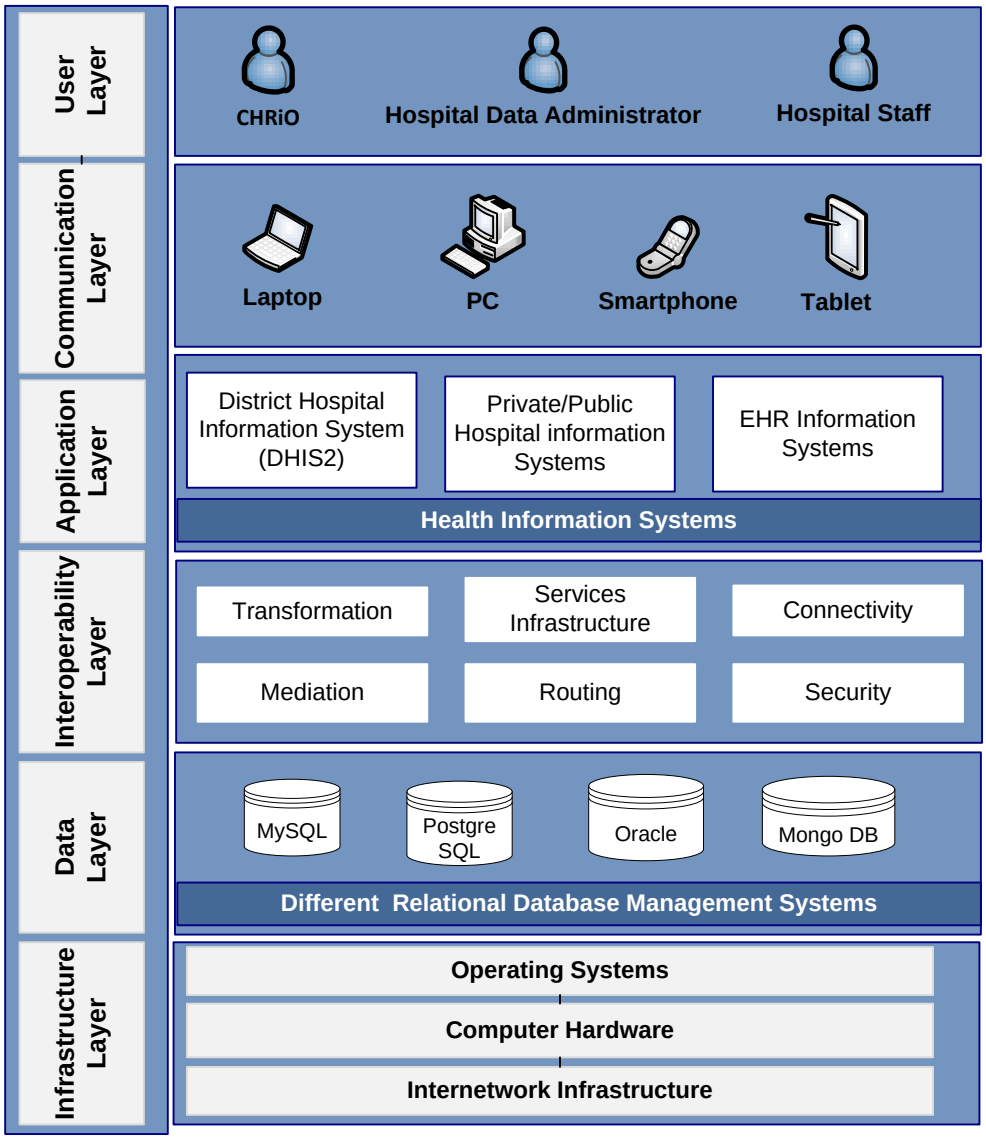


Figure 15: **The layer architecture for health information systems**

The database and health information systems layer reside at the service client and service providers' ends. The database information would be made available to requestors through the data access layer, while the health information systems layer is where the interoperability rules that determine how data is created, stored and accessed are implemented. The database technologies and interoperability implementations vary across service requesters and the service providers. As a result of the service-oriented architecture being implemented, external access to these databases is provided via services. The services are implemented

using web service technology. The choice of web services technology is due to its support for open technologies and protocols such as HTTP (hypertext transfer protocol), XML (extensible markup language), and WSDL (web service definition language) among others. These open technologies provide a desired level of interoperability and easy means of access. The web services provide functionality to access the databases while implementing the integration rules at the backend. The individual services are registered in a repository from where they can be accessed. Users can access services directly in repositories; in this case, access will be through a middleware. The data exchange bus is the middleware between service requestors and the services providers.

Since the requestors cannot consume the services directly from the service repository, the bus performs the task of mediating between the requestor and provider. The WDEB performs service orchestration. Service consumers cannot access the Web Services directly so a further interface is required.

Service consumers request services via an array of devices ranging from desktop computers and laptops to mobile devices such as smart phones and tablets. A universal mode of access is required for all these devices hence the choice of a web interface. Most modern devices contain a web browser that can be used to access web content irrespective of operating platforms. As a result, a web server is required to provide a presentation layer in form of web pages to the consumers via which they can access the services and view results.

This choice of architecture is effective because it provides access to a wide range of consumers and devices as a result of the ubiquitous technologies employed in the presentation layer while also providing a high level of separation of concern. However, the consumers are loosely coupled to the individual services. This provides flexibility because the service providers can switch database technologies or business logic implementation without affecting the consumer as long as the exposed services conform to a predefined contract that the consumer is accustomed to. Implementation of services by providers is also relatively cheap as services can be built on existing infrastructures without restructuring the whole system.

Infrastructure Layer consists of the computer hardware (Servers, desktops, peripheral devices, UPS, etc.); civil infrastructure designed for (server rooms, control centers, etc.) and internetwork infrastructure (Structured cabling, switchers, routers, fiber optic channels, etc).

The Data Layer is composed of different proprietary relational databases i.e. MySQL, Oracle, PostgreSQL and others; the integration layer exposes functions to both external and inter users. The Application Layer integrates functions into modules which is made available to communication carriers define in the communication layer. Users Layer represent users of the system.

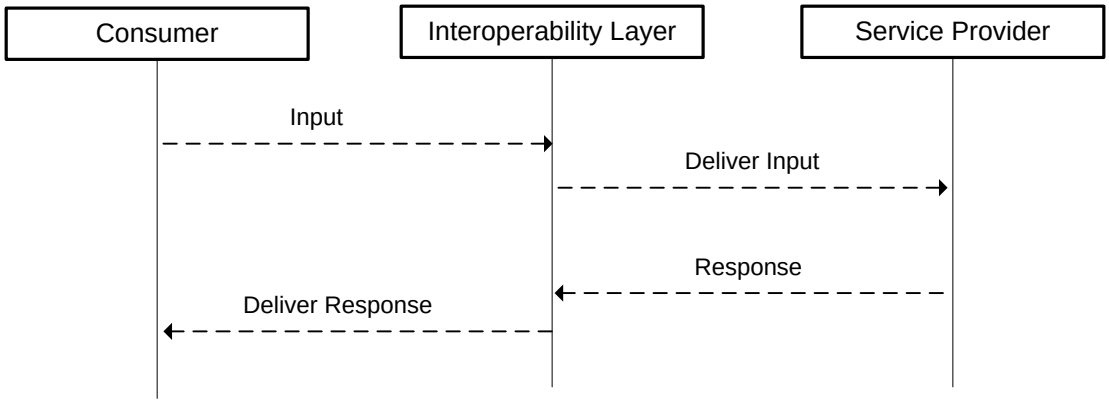


Figure 16: **Usage of the Interoperability Layer by the Health Information Systems**

4.1.5 The Interoperability Process of the Web Service Data Exchange Bus

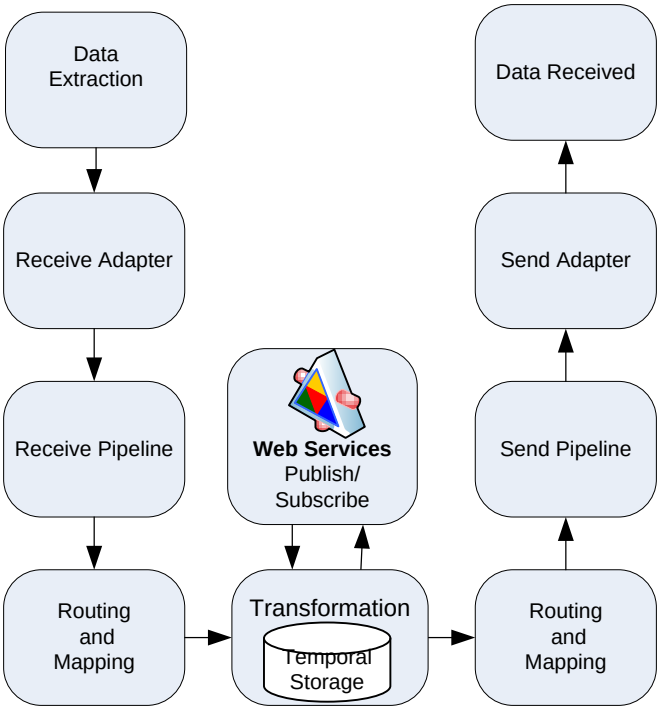


Figure 17:Interoperability process

4.1.6 The interoperability process consist of following main stages ;-

i **Data Extraction** – The prototype will allow the user to load a text file which has data values that are found in data set or reporting tool in DHIS2. The text file would contain aggregated values from the tallying sheets of a certain health facility and which had been collected during a specific period.

In data extraction process, the bus supports communication protocols and methods of connecting to a remote HISs databases. Connection to the remote applications from the bus creates a pathway through the Internet to a specific database and a query statement is executed on the remote side or on the partner HIS side. The database results are captured transmitted to a temporal storage on the bus. The data is relayed on the Receive Adapter and stored on the pipeline/temporary.

ii **Routing and Mapping** – The data values harvested from a text will be statically or manually mapped to data sets and data elements loaded in a real-time way from the DHIS2, the same will occur to data accessed from the remote Hospital Information Systems through public network or intranet.

Once the mapping of the data values from HISs and data elements from DHIS2 is statically mapped initially, the exchange bus automatically retains the mapping configurations to the remote applications and also mapping configurations between the values and elements as well. The data source and data destination configurations are automatically retained where the process of mapping and routing is done once and configurations are stored in the prototype.

iii **Transformation** – The received data in the bus is in different formats, plain text and data from different databases is temporary stored where the it is copied to a JSON parser that translates it to JSON format. The bus which is connected directly to DHIS2 web services, it mediates applications which cannot interface or directly connect to the published web services.

After the data is transformed to JSON objects the specific web service is invoked that enables DHIS2 to receive and store the data objects.

4.1.7Data Exchange Service Bus Data Flow

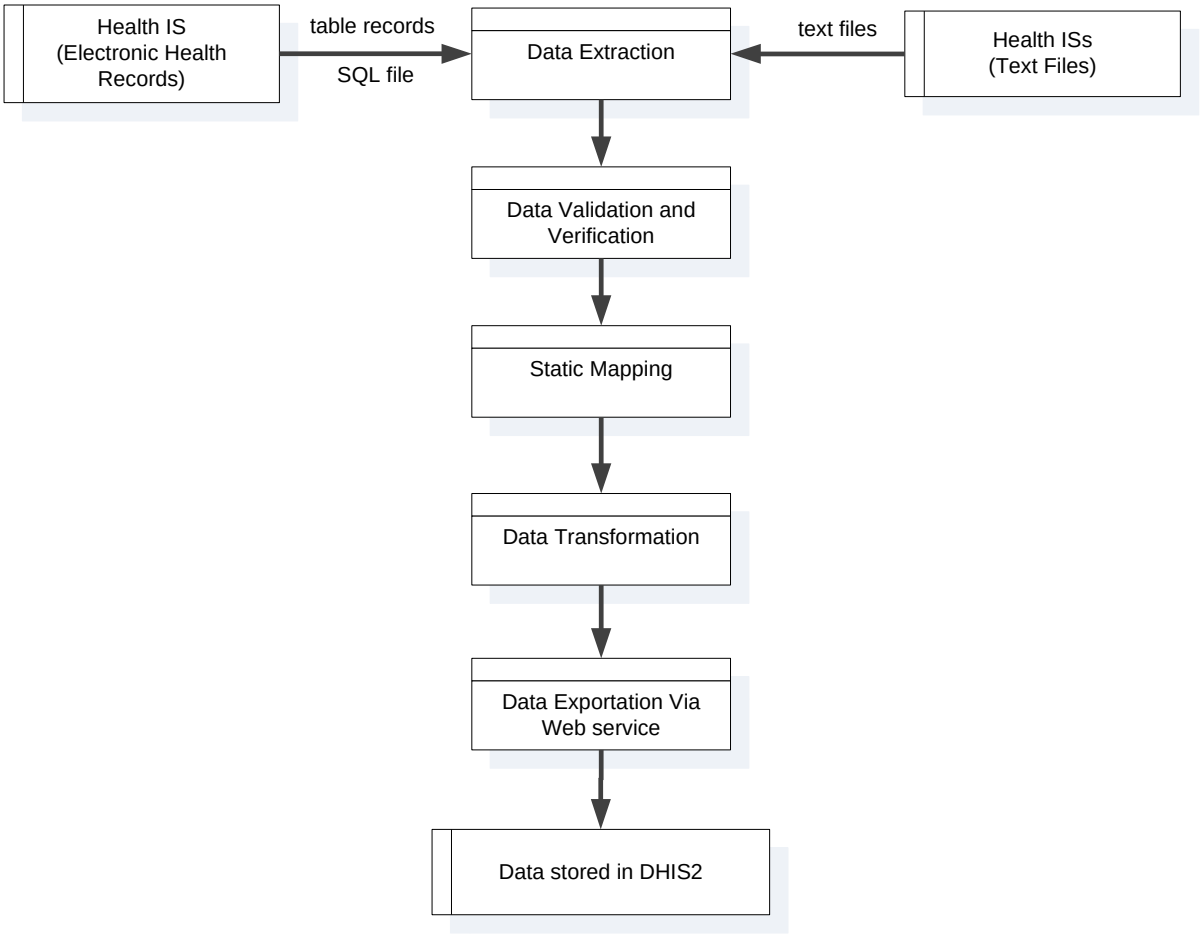


Figure 18:Data Exchange Service Bus Data Flow

The main data sources of the bus are health facilities which have information systems that maintain the health records of the patients and also facilities which submit reporting data while typed on a plain text file in a spreadsheets files.

Firstly, the data is obtained from the health facilities in form of text files or through online access from the Hospital information System diverse databases.

Secondly, the data from facilities is checked for validity and completeness as it is received into the bus.

Thirdly, the harvested data values from the information systems are statically mapped to the data elements from the service provider.

Fourthly, the data is translated to the canonical format which is recognizable by the recipient application.

4.1.8The Web Data Exchange Bus Activity Diagram

Activity diagrams show the sequence of activities in a process, including sequential and parallel activities, and decisions that are made. The above activity diagram is for CHRIO or Hospital use case and shows the different possible scenarios.

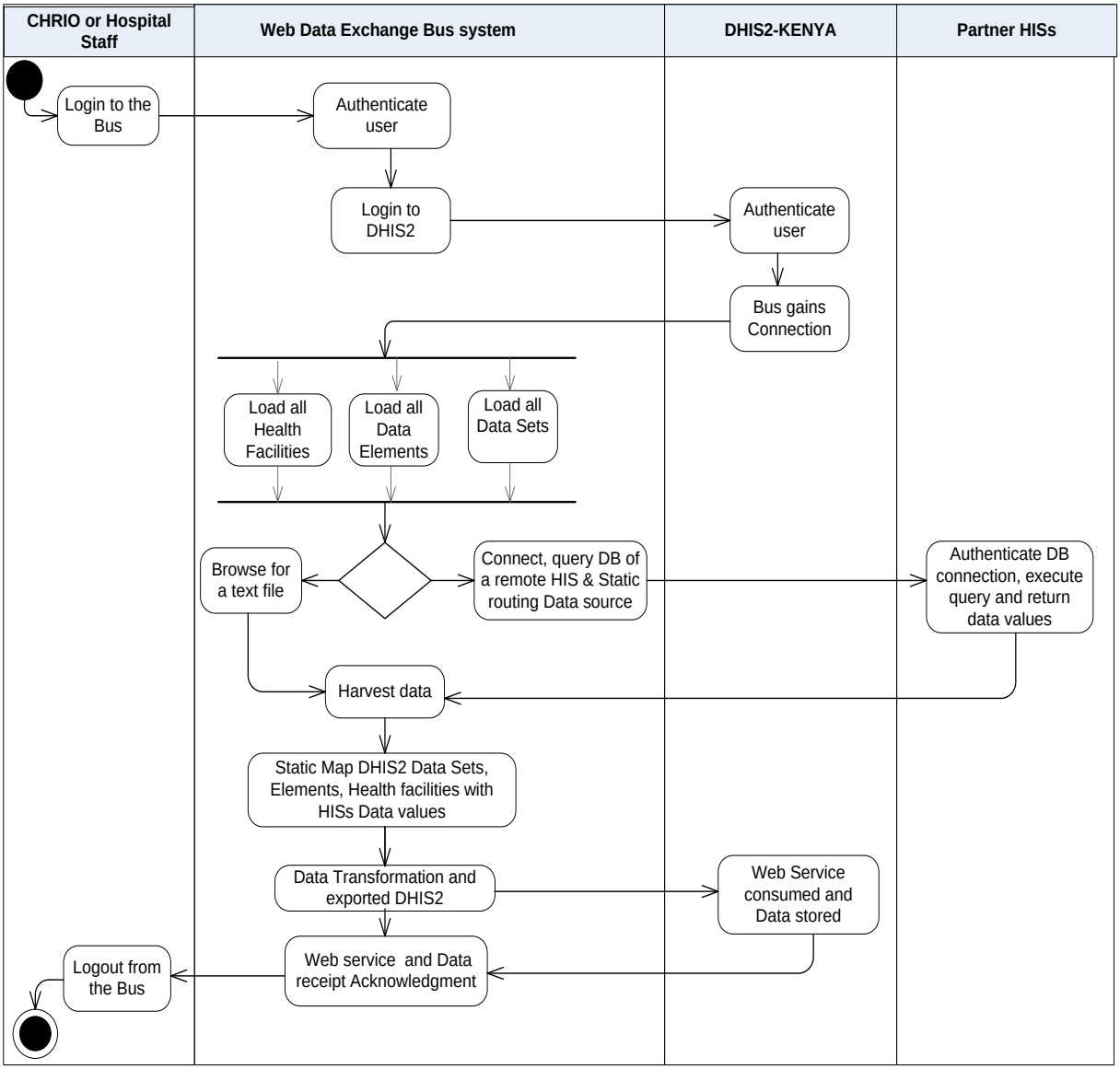


Figure 19:Data Exchange Service Bus Activity Diagram

Firstly the activity in the prototype starts by the logging in, the user account must be active or the user login will not be authenticated. Secondly after gaining access into the system, the user must login to DHIS2 through the bus. Thirdly, the bus will gain access and connect to DHIS2 resources and pull important data sets and elements :

i. Lists all the health facilities that report or submit morbidity data to DHIS2

ii. Lists all reporting tools, the Data Sets forms

iii. Lists all data elements

Fourthly, having established a connection and loaded the elements, the user loads data from file sent from a health facility or connect to application and run an SQL query statement.

The connections to remote databases are statically entered, thus routing is static however configurations are saved in the prototype dynamically.

The data values are loaded from file or remote databases by clicking Harvest button.

Fifthly, In the left pane of the Bus window will display data values and in right pane will display data elements, the user will map the data values to the DHIS2 elements for a specific data set. The mapping is done manually but the configurations are automatically saved in the Bus whereby the subsequent time the user requires to do a monthly reporting s/he will use the previous configurations.

Finally after mapping and routing, the user invoke the transforming of the data and executing of the web service by the bus where DHIS2 will acknowledge delivery of the data.

4.1.9 Canonical Data Format Transformation

The integration architecture of the data exchange bus needs to translate or convert any received data to a common data format which can be understood by all other systems that are connected to the bus. The common format is referred to as canonical, eXtensible Mark-up Language (XML) and JavaScript Object Notation (JSON) have been the standard canonical models or formats used to allow interchange of messages between heterogeneous applications. The prototype will use JSON canonical as the standard format. After the data is validated, it is transformed and routed to the destination application for storage, DHIS2.

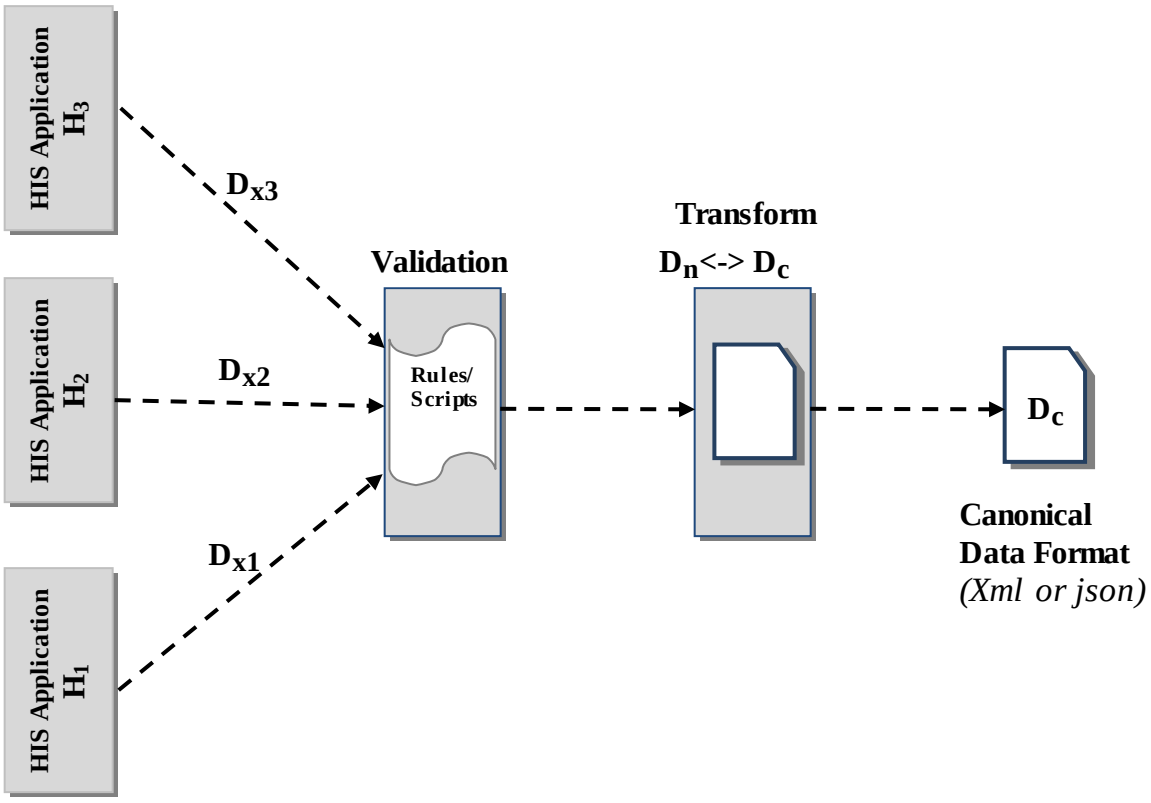


Figure 20: **Transformation process**

Applications are loosely decoupled since they can communicate or exchange messages with other applications notwithstanding the data formats that they support.

4.2.0Non-Functional Requirements

i. **Flexibility** – the prototype has the capabilities of integration technologies toward rapid adjustments. For example modifications of the software, with minimum effort, operational and functional capabilities in various computing environments.

ii. **Real time** - ability of the integration technologies to support operations as they happen at the same time or access data elements on a remote data storage in their up-to-date form.

iii. **Reliability** - the techniques and protocols which are practiced in the integration technologies to ensure all transmitted data

iv. **Reusability** - to the ability to use existing information system components or software solutions to develop new applications in the health information system domain. Reusability reduces the time and cost of implementation. It has a significant role in system integration and the results are more maintainable and flexible system.

v. **Performance**- refers to the performance of the system, the integration system provide integration however the performance of overall integration solution is satisfactory.

vi. **Maintainability** - ability of bus components and software applications to allow changes without causing any problems in other systems. Integration technologies for bus easily maintained.

vii. **Portability**– the bus is easily executed on different platforms. Portability is related to the concept of standards and provides an important role in the cost effectiveness of information systems.

viii. **Scalability**-the bus has the ability to supply high performance to accommodate a growing future loads and increasing demands.

ix. **Heterogeneity**– has the capability of interoperating of legacy and new health information system through the availability of proper programming language and operating system platforms.

System Design and Implementation

We developed the Web Data Exchange Bus using PHP language and hosted it on a Apache Web server in Linux (Kubuntu) environment. The prototype doesn't have a database for storing data since data would be loaded from service consumer and provider then the data is destroyed when the session ends.

We also hosted a Hospital Information System locally which has database for managing electronic health records of patients.

Below is the Login user interface of the prototype

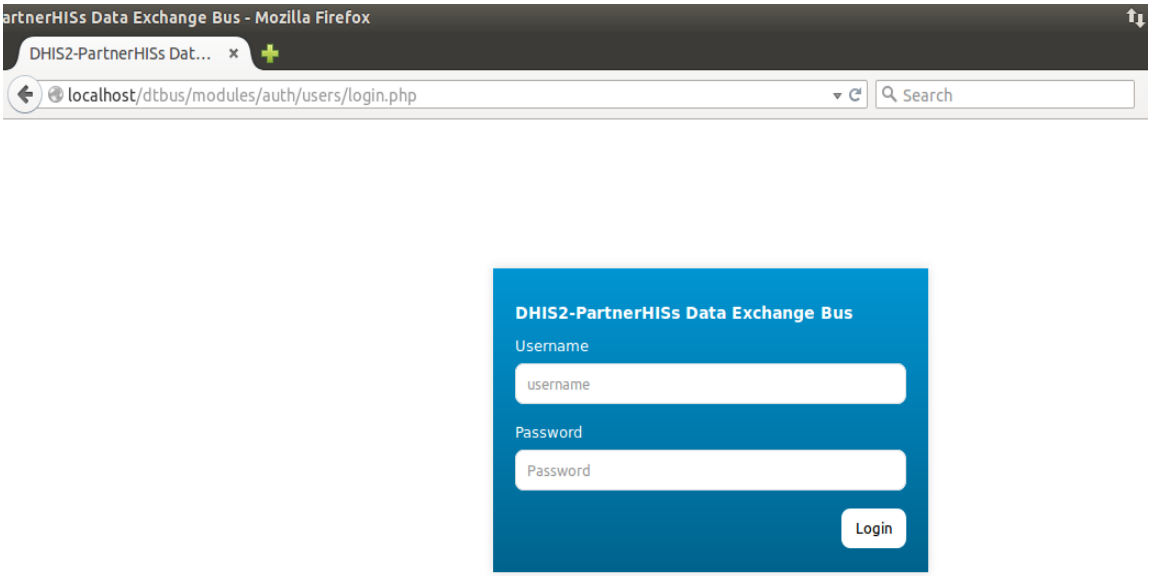


Figure 21:Login user interface of the prototype - BUS

Below is the Login user interface to the recipient system - DHIS2

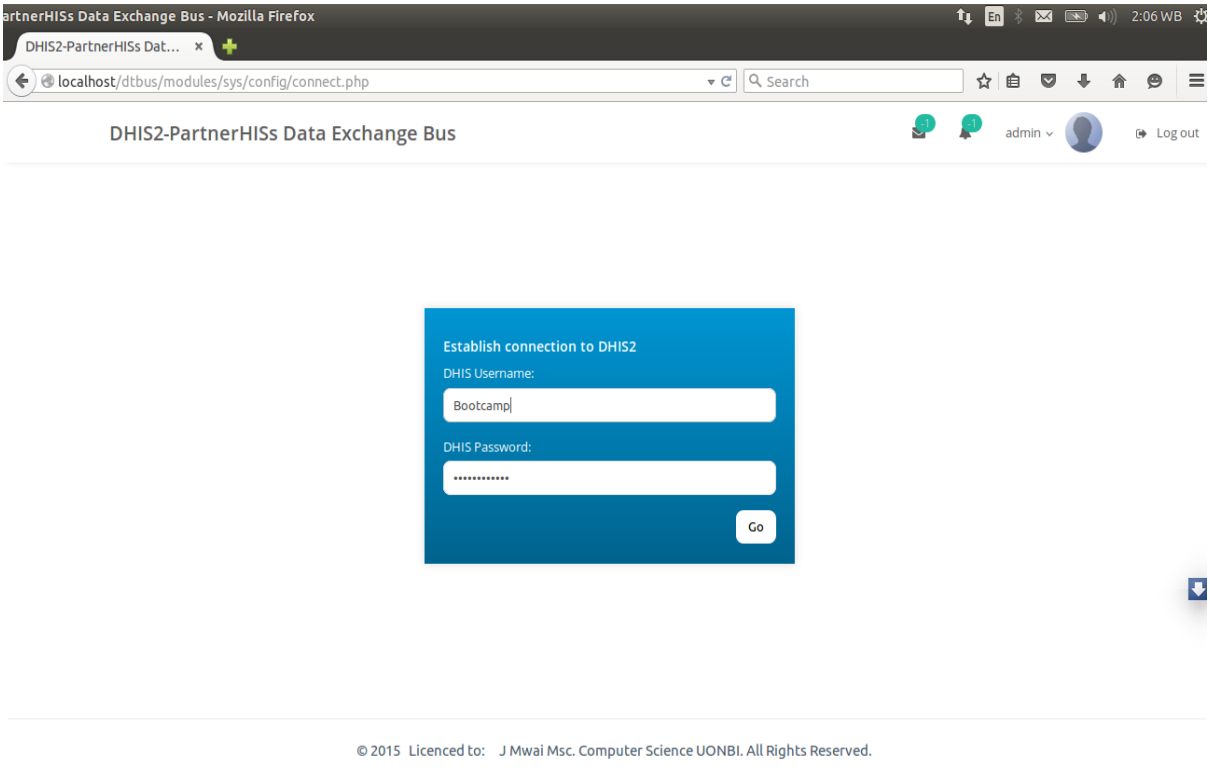


Figure 22:Login user interface to the recipient system - DHIS2

Below is interface of the prototype after gaining connection to Service Provider DHIS2

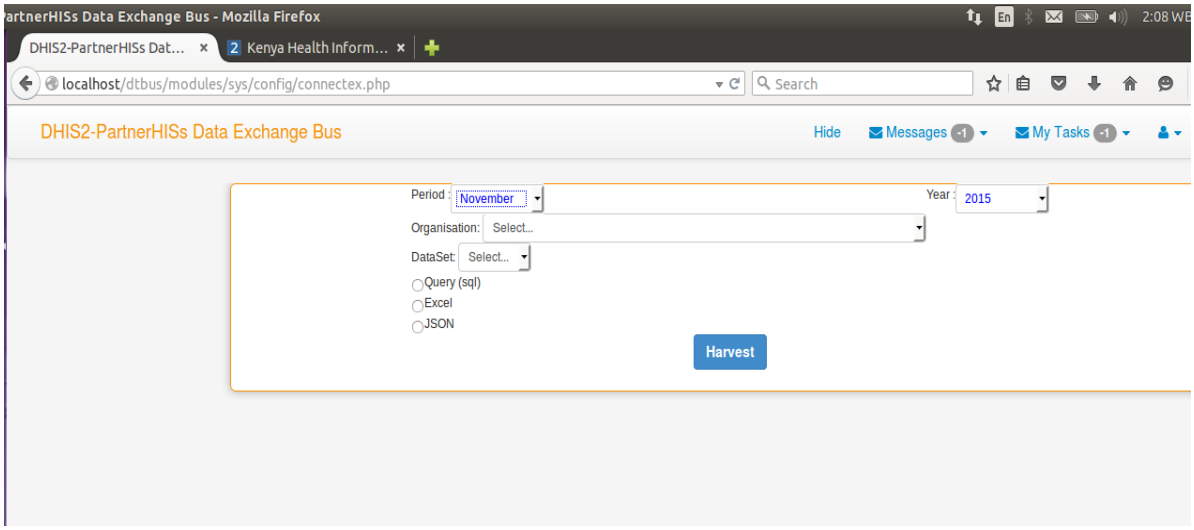


Figure 23:Interface of the prototype after gaining connection to Service ProviderDHIS2

Below is the prototype after gaining connection to Service Provider DHIS and all listing health facilities

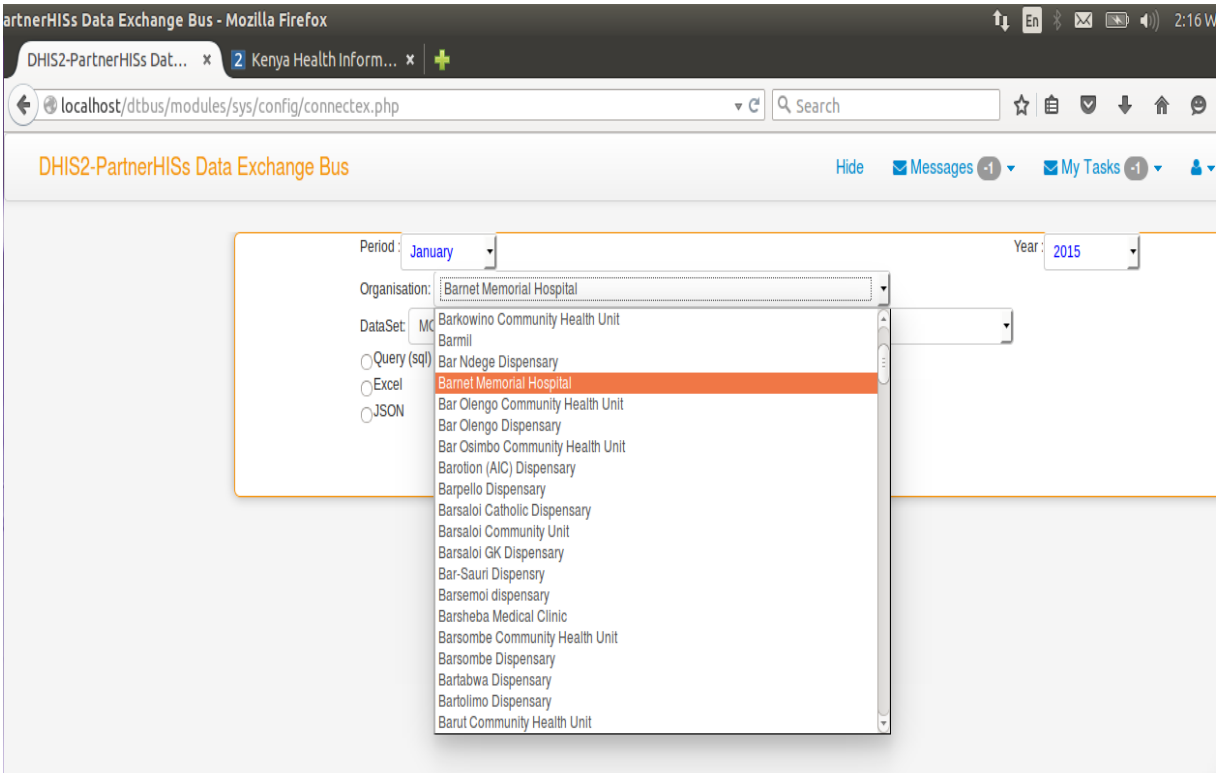
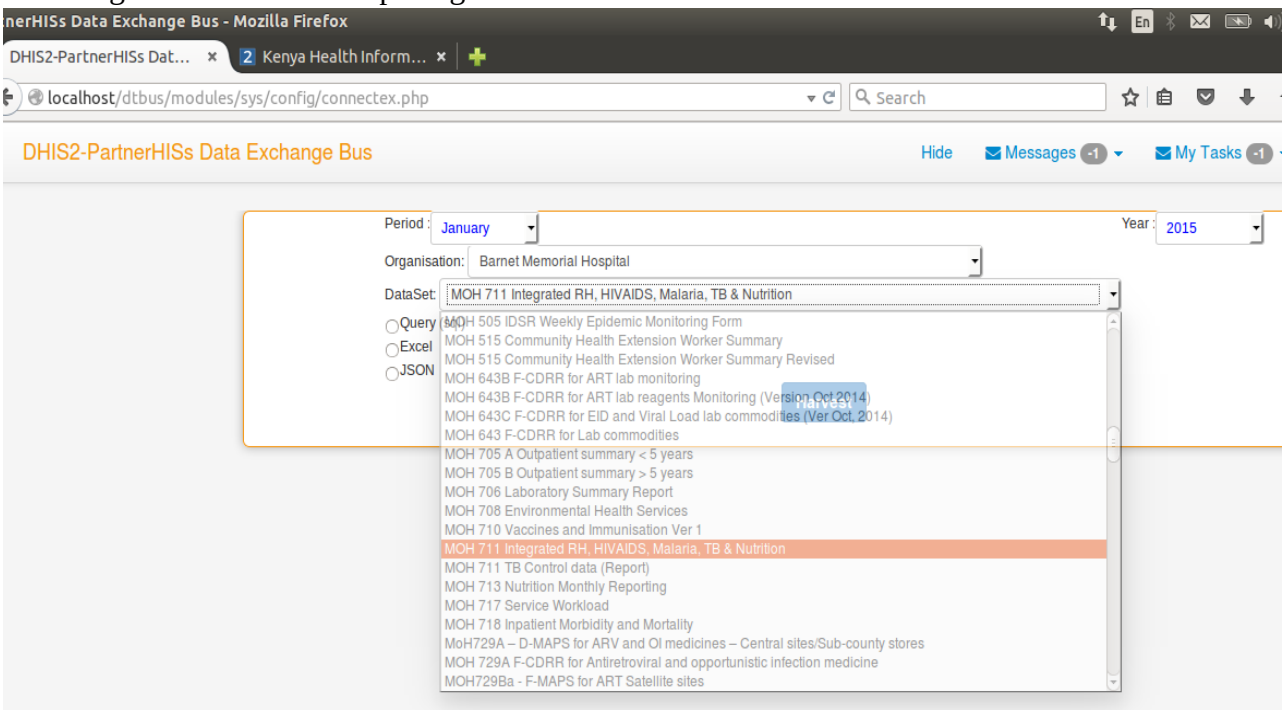


Figure 24:Interface of prototype and listing health facilities

Below is interface of the prototype after gaining connection to Service Provider DHIS2 and listing all Data Sets or Reporting Tools



Interface of the prototype affording user options for sourcing a data file

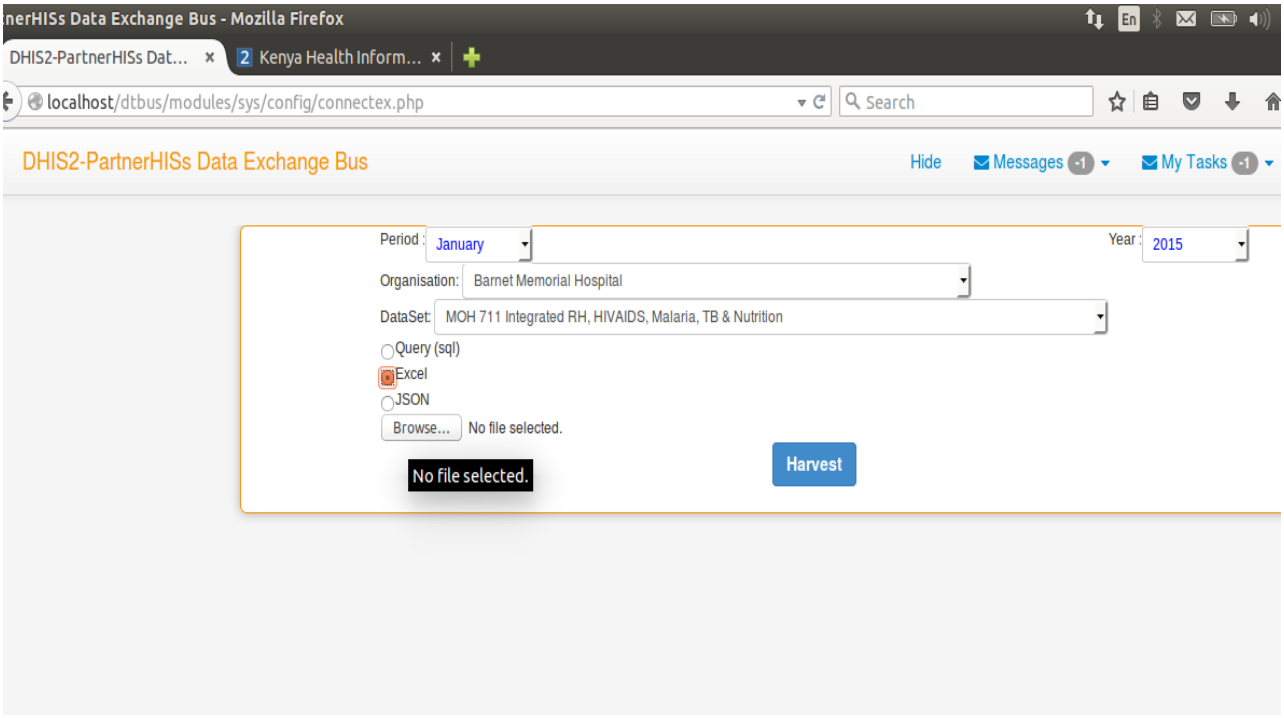


Figure 25: Interface affording user options for sourcing data

Interface that affords user option to load a spreadsheet with data to be consumed by another HIS.

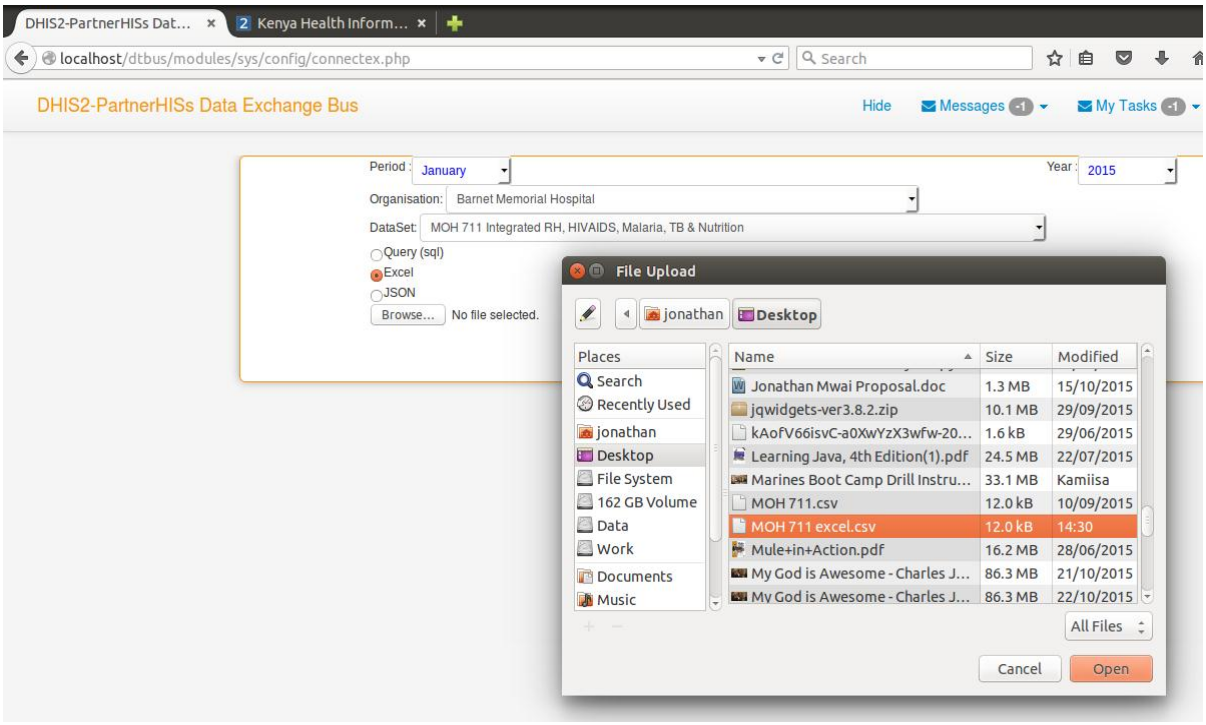


Figure 26:User options to load a spreadsheet with the values to be consumed by another HIS.

Interface after harvesting HIS data values from the file but not mapped them to DHIS2 data elements

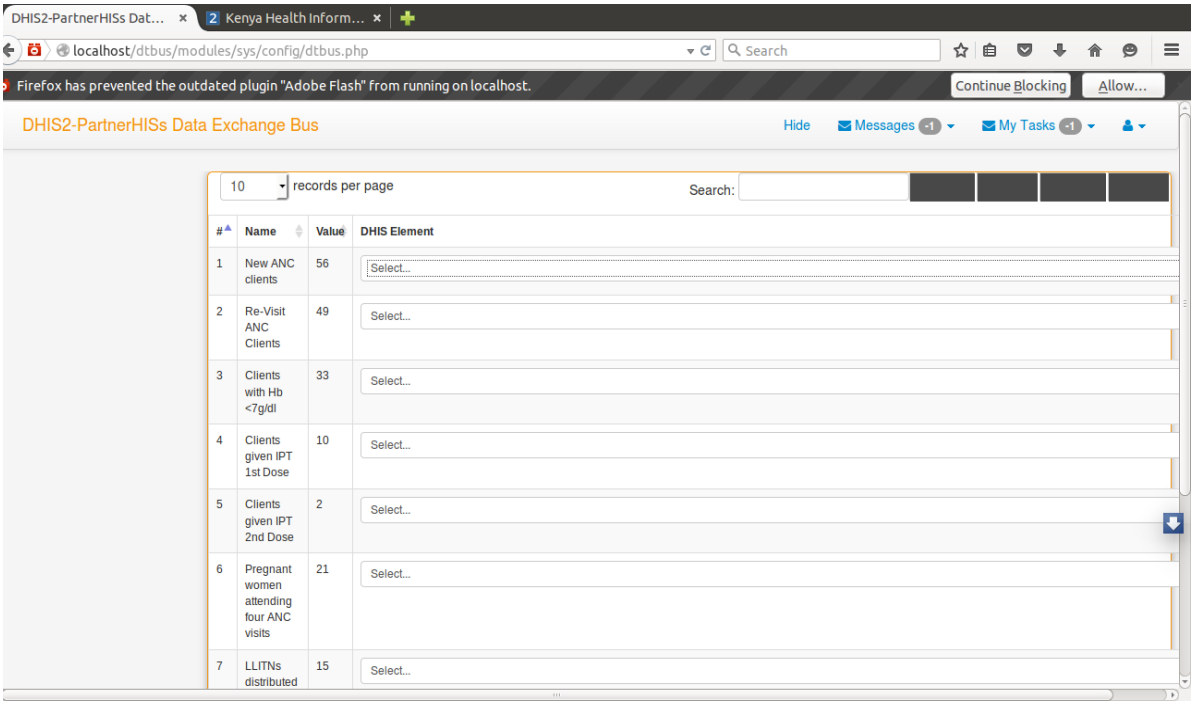


Figure 27:Harvested HIS data values but not mapped them to DHIS2 data elements

Interface the user mapping HIS data values to the DHIS2 elements

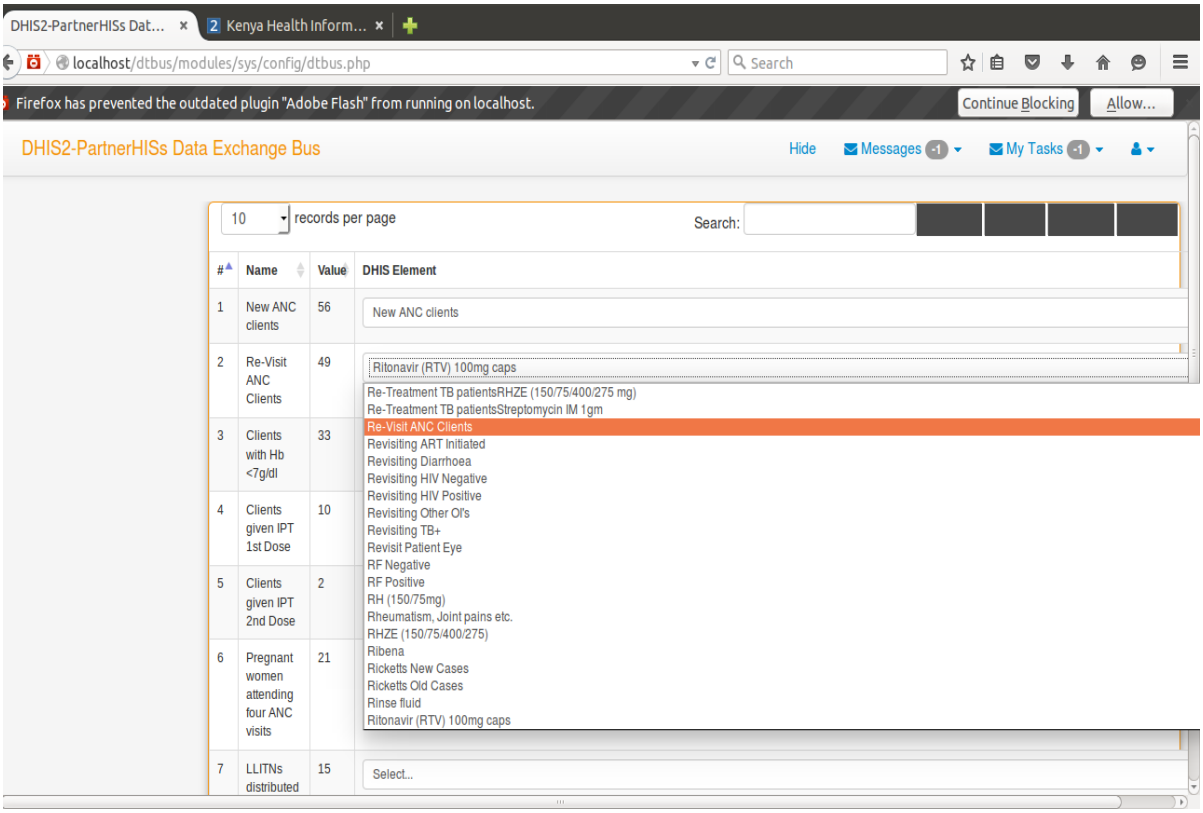


Figure 28:Interface for mapping data values to the data elements.

Interface for user to specify location of the remote host, database connection credentials

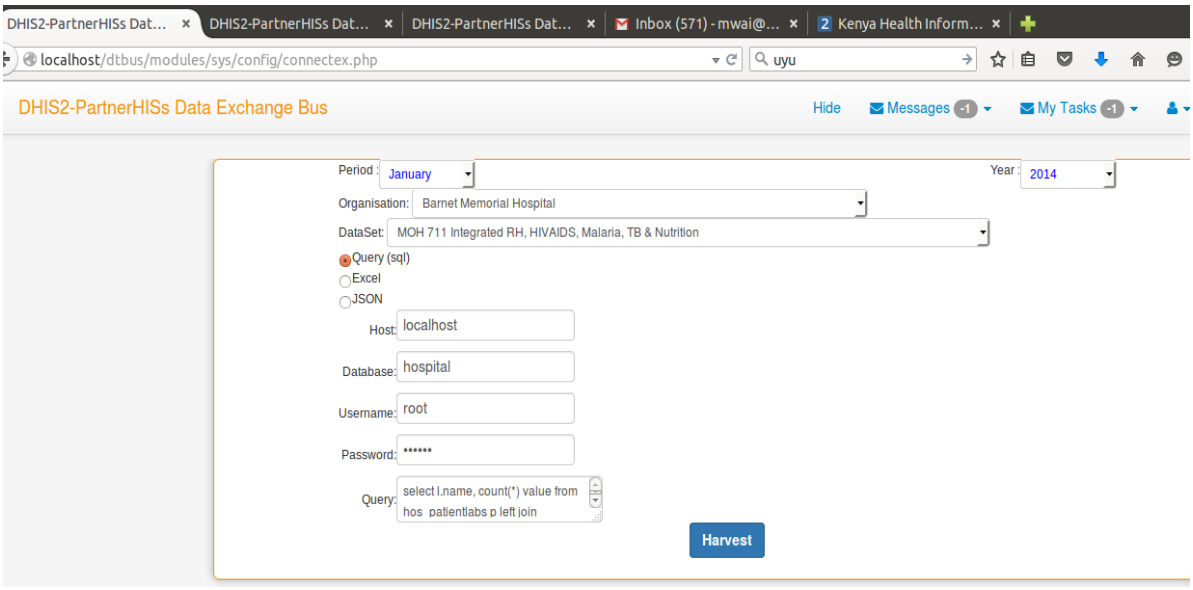


Figure 29:Specify location/ip-address of the remote host, database connection credentials

Before data was received by the recipient DHIS2 system

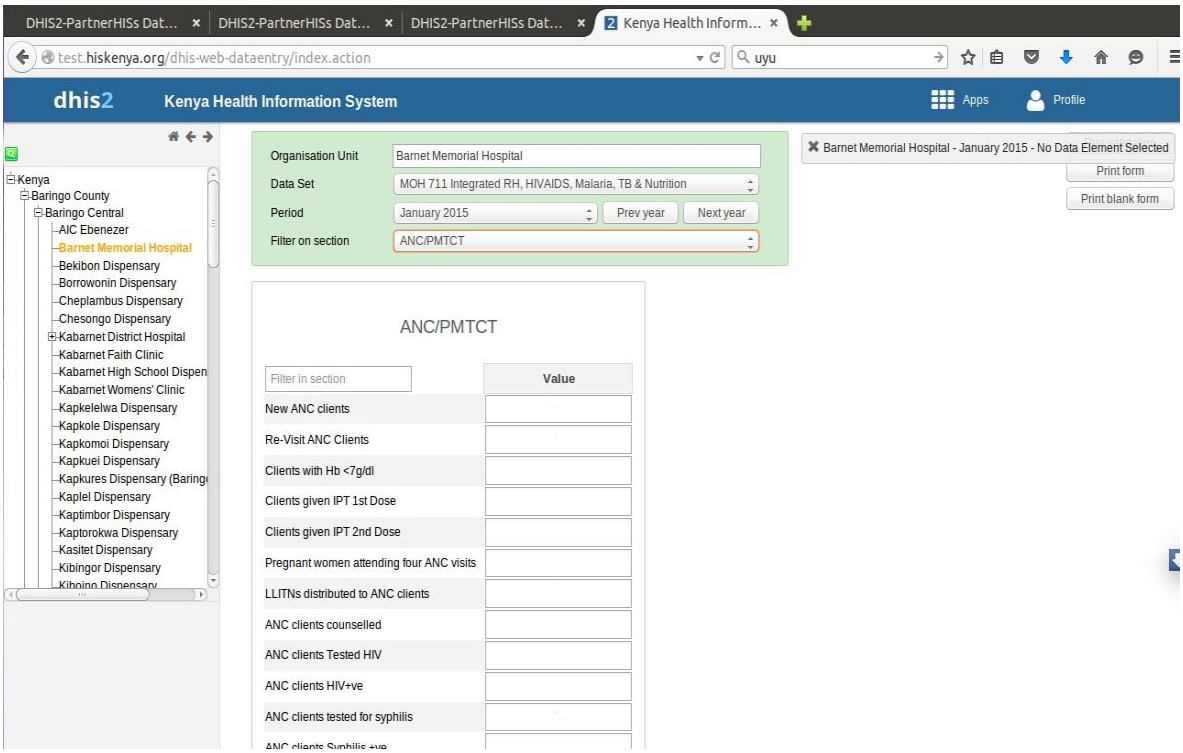


Figure 30:Before data was received by the recipient DHIS2 system
After data was received by the recipient DHIS2 system

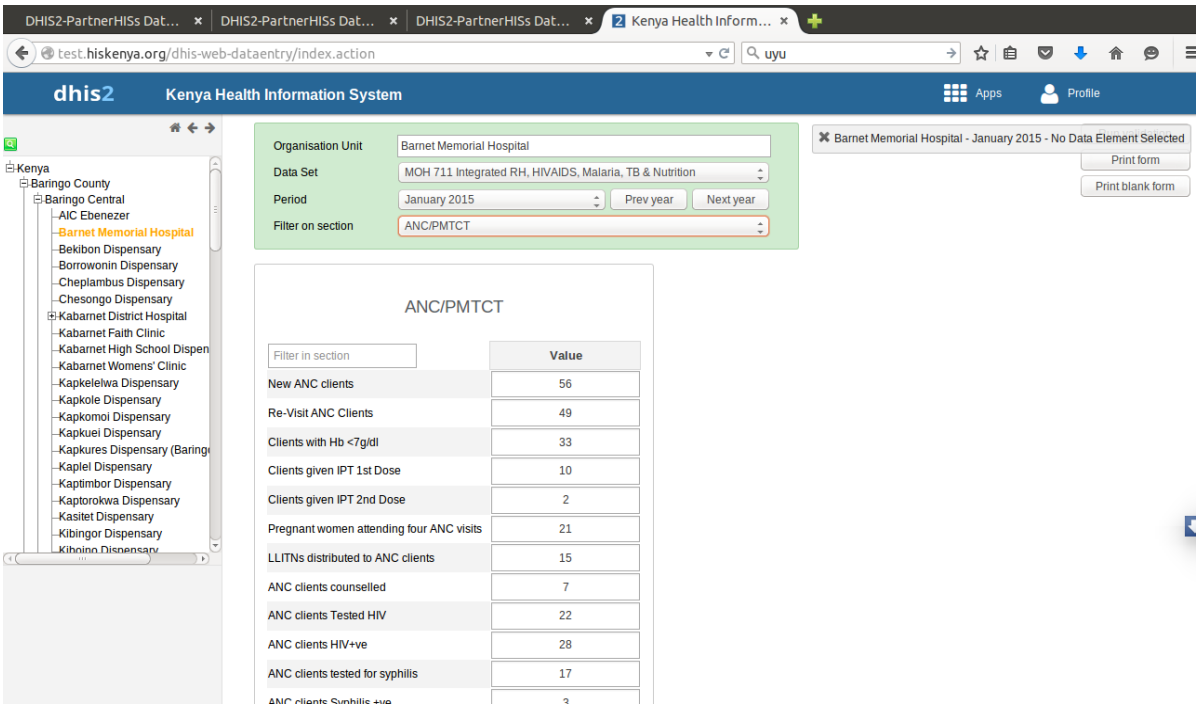


Figure 31:Data received by recipient system DHIS2

5.0 CHAPTER FIVE: RESULTS AND DISCUSSION

In this section we analyze the extent to which the project concerns are satisfied by the developed solution.

The Kenya Ministry of Health and other health stakeholders highly depend on DHIS2 in management of routine health services delivery and making of informed decisions based on analysis of information collected from all health facilities in the country. The invaluable data from almost all health facilities is collected mainly through manually filling of reporting tools or data sets sheets in the facilities and later the sheets collected at County level where the data is keyed into DHIS2 system by the County Health Records Information Officers(CHRIOs).

By implementing the WDESB for integrating incompatible systems together, through the bus data from HIS which are web enabled is imported to DHIS2 more efficiently and effectively solving problems caused by manual data management. By use of the WDESB the data are delivered to DHIS2 on time and complete without errors since it is obtained directly from the HIS and on time. The manual method of data collection in the health facilities by filling of the tallying sheets later recording it on reporting data sets sheet has been prone to delays in delivering the sheets to the county level offices and also data incompleteness due to errors when recording on the sheets.

5.1 There are desirable characteristics in interoperability achieved by the system.

i. Facilitate interoperability

The prototype facilitates interoperability by providing a mechanism through which disparate HISs communicate. The prototype provides a physical communication channel (bus) through which the disparate systems may communicate.

Consider two points of systems (H1 and H2) that wish to communicate reporting aggregates via the bus. The physical part of the communication is provided by supplying an endpoint for system H1 and H2 to send data to (the interface component). The users are authenticated by the security module and then the data is extracted, transformed then routed to a mediator that is able to send the message to system H2.

ii. Security

The System ensures only authorized users can access the services in the bus. Data confidentiality and integrity are of great concerns when dealing with health information, this is taken care of in the bus by the authorization and authentication security module.

iii. Re-usability

The very key processes in the bus that is the mapping of data values and the elements and retaining of service requestor and provider configurations are dynamically saved after the initial configuration is done. The bus retains initial connection to service provider and it also provides real-time access and use of the up-to-date data sets, elements and services available at the service provider

With the proposed system, the hospital staff at the health facilities and the CHRIO at the county level can now submit the monthly reports directly to DHIS2 as long as their HISs are web enabled or through an SQL data file or spreadsheet file, the main function of the data service bus is to mediate HISs with no capacity to consume Web Services registered within the service repository which is discoverable via the service bus.

Reports on outbreak of diseases can easily be traced from databases from hospitals based on reported cases as well as other statistics can be obtained as well.

Researchers can also now use information obtained from hospital databases easily and in real-time. Using SOA with web services makes it easy for heterogeneous HISs to be integrated and interoperate. Services created can be reused in multiple ways and also new services and applications can be created quickly and easily used with a combination of new and old services.

5.2 Prototype Testing

5.3 Comparison between Web services and API Integrating Approaches

API (Application Programming Interface) has been a major methodto integrate two different systems or IT services.[31]An application programming interface (API) is a set of routines, data structures, object classes and/or protocols provided by libraries and/or operating system services in order to support the building of applications or exchange of data objects between two applications.

A Web Service is defined by the W3C as "a software system designed to support interoperable machine-to-machine interaction over a network." The difference is that Web Service almost always involves communication over network and HTTP is the most commonly used protocol. Web service also uses SOAP, REST, and XML-RPC as a means of communication. While an API can use any means of communication e.g. DLL files in C/C++, Jar files/ RMI in java, Interrupts in Linux kernel API etc

The WDESB is a software infrastructure for Service Oriented Architecture implementation we have used as a medium for connecting disparate the health information systems. We will compare between the major methods of integrating systems, API and Web Services.

Web Services	**Application Programming Interface (API)**
Loosely coupled – works for multiple connections	Tightly coupled – works for one-to-one connection
Easy to implement security management	lacks security management
Have well established standards: i. Messaging Standards- SOAP, ii. Description and discovery Standards - UDDI and WDSL.. iii. Security Standards - Security Assertion Markup Language (SAML). iv. Management Standards - Web Services Distributed Management v. Transport Standards - XML or JSON.	Lacks standards due to systems being greatly different from each other and proprietary
Programming code is organized as from development and deployment perspective allowing service interaction	Programming code is in form of object classes, routines, does not support SOA,
Services are distributed on networks and they are integrated with each other via a central service registry, thus services are discoverable when queried by any client services/applications	Resources are not published at a central registry and not accessible through network/not distributed

Table 1:Comparison between Web services and API integrating approaches

5.4 Conclusion

In health information systems, the importance of addressing interoperability issues among existing systems is widely recognized. A crucial aspect is to allow health information systems to exchange information in a pervasive andreliable way, even if these data are distributed in technically and geographically different health information systems.

In this study, we identified the various approaches for integrating applications, various technologies and standards for aiding interoperability between the incompatible health information systems.

After reviewing the literature we identified the aspects in system development that prevent interoperability between the different health-care information systems. In order to promote the exchange of data between the HISs we identified the relevant interoperability technologies and standards. Having looked at application of the technologies and standards in three related works we acquired knowledge of how to implement the prototype to enable HISs and DHIS2 to interoperate.

In this research we proposed a prototype based on SOA architecture and Enterprise Service Bus that takes advantage of adoption of the established standards. A service-oriented architecture facilitates an ESB which acts as the communication backbone thus creating the option for integration. We also studied barriers that prevent systems to communicate or exchange information and discussed solutions to the impedances: Semantic interoperability, Technical interoperability, and Process interoperability.

We looked at the processes involved in system interoperability and functions of components of an ESB which is a secure and efficient option of integration.

After reviewing the literature we acquired knowledge of how to implement the prototype to enable HISs and DHIS2 to interoperate.

5.5 Recommendation for Future Work

Future work will research on implementing semantic interoperability between HISs ensuring data values in one application have the same meaning in the other application consuming the data. Developing a health data semantic framework that defines exact data elements to be used while developing Health Information Systems and a common meaning that the elements should have or mean for example data elements like Sex or Gender might not have same mean. The framework would state clearly data elements to be used universally in all HISs and the elements have the same meaning to promote semantic interoperability between the HISs in a specific region like in a country.

References

[1] Institute of Electrical and Electronics Engineers. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard ComputerGlossaries. New York, NY: 1990.

[2] O'Brien, L., Bass, L. and Merson, P. (September, 2005). Quality Attributes for Service-Oriented Architectures.

[3] Health Information and Quality Authority, (July 2013),Overview of Healthcare Interoperability Standards, Published by Health Information and Quality Authority. Available from: www.hiqa.ie Accessed on: 20th Jan 14.

[4] International Telecommunication Union, (2012), E-Health Standards and Interoperability. ITU-T Technology Watch Report.2012 [Online]. Available from: www.itu.int/en/ITU-T/techwatch/Pages/ehealth-standards.aspx. Accessed on: 29 January 2014.

[5] Health Information and Quality Authority, (2011), Developing National eHealth Interoperability Standards for Ireland: A Consultation Document.

[6] Sprivulis, P., Walker,J., D Johnston, E Pan, J Adler-Milstein, B Middleton, et al., (2011), The Economic Benefits of Health Information Exchange Interoperability for Australia, Australian Health Review.

[7] Health Information and Quality Authority, (2012), Guidance on Messaging Standards for Ireland, published by Health Information and Quality Information.

[8] Health Level Seven (HL7), (2011), HL7 elearning notes. Available from: http://www.hl7elc.org/campus/. Accessed on: 11 February 2014.

[9] Sartipi, K., and Yarmand, M. (2008), Standard-based Data and Service Interoperability in eHealth Systems, published by Canada Infoway – Health Standardization Organization.

[10] AL-Khawlani, M. (2004), Web Services: A Bridge between CORBA and DCOM, Published Journal of Science & Technology Vol. (9).

[11] Sun Microsystems Inc, (June, 2004). Assessing Your SOA Readiness, Available from http://www.sun.com/software/whitepapers/webservices/soa_ready.pdf Accessed on 11 February 2014.

[12] Albreshne, A., P. Fuhrer, and J., Pasquier, (2009), Web Services Technologies: State of the Art Definitions, Standards and Case Study, A Working Paper.

[13] Zhang,J., Ewins, D. and W. Xu, (2007), System Interoperability Study for Healthcare Information System with Web Services: A Case Study for System Interoperability Concern in Healthcare Field, Journal of Computer Science 3 (7), 515-522, 200

[14] Rodrigues, G. J., Cunha C. R. and Morais, E. P., (2011), A SOA based architecture to promote ubiquity and interoperability among health information systems(Portuguese), Creating Global Competitive Economies: A360-Degree Approach.

[15] Valle, E. D. *et al.*, (2008),The Need for Semantic Web Service in the eHealth: Towards semantic interoperability based on Semantic Web Services

[16] L.P. Chan et al, (2005). Web Service Implementation Methodology, Public Review Draft : OASIS, Available from: cation: 7 http://www.oasis-open.org/committees/ documents.php?wg_abbrev=fwsi 8, Accessed on: 10 March 2014.

[17] Kester, Q. A., Gyankumah, G. N., and Kayode, A. I. (2012). Using Web Services Standards for Dealing with Complexities of Multiple Incompatible Applications International Journal of Information Technology, 4.

[18] Qusay H. Mahmoud, (2005). Service-Oriented Architecture (SOA) and Web Services: The Road to Enterprise Application Integration (EAI) .Available: http://www.oracle.com/technetwork/articles/javase/soa -142870.htmNovember 10th, 2015

[19] Haines, M. (2007) "The Impact of Service-Oriented Application Development of Software Development Methodology," Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS).

[20] Papazoglou, M. P.,(2007), What's in a Service?, INFOLAB, Tilburg University, Dept. of Information Systems &Management, Tilburg 5000 LE, The Netherlands.

[21] Jboss, (2006), Why ESB and SOA?,http://docs.jboss.org/jbossesb/whitepapers/ WhyESB.pdf last accessed on November 10th, 2015

[22] Vollmer,K.,(2011), The Forrester Wave™: Enterprise Service Bus, Application Development & Delivery Professionals Consortium.

[23] http://msdn.microsoft.com/en-us/library/ff647958.aspx#intpattch05_pointto point connection, last accessed on October 8th, 2015

[24] http://www.poltman.com/en/technical-information/eai/topologies, last accessed on October 15th, 2015.

[25] http://www.scis.ulster.ac.uk/~zumao/teaching/COM720/readings/reading10.pdf last accessed on September 28th, 2015.

[26] Chappell, David (2004). Enterprise Service Bus. O'Reilly Media, Inc.[27] Arsanjani, A., Borges, B., and K. Holley, (2013), Service-oriented architecture: Components and modeling can make the difference. Web Services Journal, 9, 1 , 34–38.

[28] Newcomer, E., and Lomow, G., Understanding SOA with Web Services. Boston: Addison Wesley.

[29] Bell, M. (2008), Service-Oriented Modeling: Service Analysis, Design, and **Architecture, Addison-Wiley**

[30] Keen M. et al, (2004), Patterns: Implementing an SOA Using an Enterprise Service Bus, http://www.redbooks.ibm.com/redbooks/pdfs/sg246346.pdf - last accessed on June 20th 2015.

[31] Chen, L.(2012), Integrating Cloud Computing Services Using Enterprise Service Bus(ESB), Published by Sciedu Press

Appendices

5.7.4 **Sample of MOH 711 Register Reporting Form**

DISTRICT FORM MOH 711_B

Appendix Five: MOH 711-Integrated Reporting Tool

REPUBLIC OF KENYA
MINISTRY OF HEALTH

NATIONAL INTEGRATED FORM- REPRODUCTIVE HEALTH, HIV/AIDS, MALARIA, TB and NUTRITION

DISTRICT: _____ MONTH: _____ YEAR: _____

No of Health facilities expected to report_____ No of H/facilities that reported_____

A: FAMILY PLANNING			NEW CLIENTS	RE-VISITS	TOTAL
1.	PILLS	Microlut			
		Microgynon			
2.	INJECTIONS	INJECTIONS			
3.	I.U.C.D	Insertion			
4.	IMPLANTS	Insertion			
5.	STERILIZATION	B.T.L			
		Vasectomy			
6.	CONDOMS	No. of Clients receiving			
7.	ALL OTHERS: (specify)				
8.	TOTAL NO. OF CLIENTS				

9.	REMOVALS:	IUCD		IMPLANTS	

B: MCH - ANC / PMCT			New	Re-visit	TOTAL
1.	No. of ANC Clients				
2.	No. of Clients with Hb < 7 g/dl				
3.	No. of Clients given IPT (1st dose)				
4.	No. of Clients given IPT (2nd dose)				
5.	No. of Clients completed 4th Antenatal Visit				
6.	No. of ITNs distributed to ANC clients				
7.	No. of ANC clients	Counselled			
		tested for HIV			
		HIV+			
8.	No. of clients	Tested for Syphilis			
		+ve			
9.	No. of clients issued with preventive ARVs				
10.	No. of infants tested for HIV	At 6 wks			
		After 3 Months			
11.	HIV+ referred for follow up	Mothers			
		Partners			
12.	No. of infants issued with preventive ARVs				
13.	No. of mothers counselled on infant feeding options				
14.	No. of partners	Counselled			
		Tested			
		HIV+			

C: MATERNITY- PMCT			TOTAL
1.	No of Women counselled		
2.	Women tested for HIV		
3.	Women found HIV +ve		
4.	No. of Women issued with preventive ARVs		
5.	No. of infant Preventive ARVs administered		
6.	Total Deliveries from HIV +ve women		
7.	No initiated cotrimoxaxole	Women	
		Infants	

E. MATERNITY / SAFE DELIVERIES		Number	
1	Normal Deliveries		
2	Caesarean Sections		
3	Breech Delivery		
4	Assisted vaginal delivery		
5	TOTAL DELIVERIES		
6	Live Births		
7	Still Births		
8	Under Weight Babies (Weight below 2500 grams)		
9	Pre-Term babies		
10	No. of babies discharged alive		
11	Referrals		
12	Neonatal Deaths		
13	Maternal Deaths		
	Maternal complications	Alive	Deaths
14	A.P.H. (Ante Partum Haemorrhage)		
15	P.P.H. (Post Partum Haemorrhage)		

D: STI		Type of visit	Females	Males	Total
1.	Urethral Discharge	Initial visit			
		Re-att			
		Referrals			

DISTRICT NAME:_____ MONTH:_____ YEAR:_____ MOH 711_B

F: PAC SERVICES		TOTAL
1.	No. of MVA	
2.	No. of D & C	
3.	No. of FP Up take	

G: TB		New	Re-att	Total
1.	No. of TB cases detected			
2.	No. of smear positive			
3.	No. of smear negatives			
4.	No. of Extrapulmonary TB patients on treatment			
5.	Total No. of TB patients on Treatment			
6.	Total No. of TB patients on Re-treatment			
7.	Total No. completed treatment			
8.	Total No. of TB Patients tested for HIV			
9.	Total No. of TB Patients HIV+ve			
10.	No. of TB HIVpatients on CPT			

H: VCT			15-24 years		≥ 25 years		Total
			M	F	M	F	
1.	VCT Clients	Counselled					
		Tested					
		HIV+					
2.	No of couples	Counselled					
		Tested					
		Both HIV+					
		With discordant					

I: DTC			Children (0-14 years)		Adults (>14yrs)		Total
			M	F	M	F	
1.	No. counselled	Outpatient					
		In-Patient					
2.	No. tested	Outpatient					
		In-Patient					
3.	No. HIV+	Outpatient					
		In-Patient					

J: CHILD HEALTH AND NUTRITION INFORMATION SYSTEM (CHANIS)				
Children Needing Follow up		F	M	TOTAL
1.	Marasmus			
2.	Kwashiorkor			
3.	Anaemia			
4.	Faltering Wt			
Others e.g. Vitamin A deficiency, etc. (Specify):				
5.	_____			

K: ART			Children 0-14yrs		Adults >14yrs		Totals		Grand Totals
			M	F	M	F	M	F	
1.	No of new patients enrolled within the month for HIV care by entry point	PMCT clients							
		VCT clients							
		TB patients							
		In patients							
		CWC							
		All others							
		Sub-total							
2.	Cumulative No. of persons enrolled in HIV care at this facility at end of the month								
3.	Number of patients starting ARVs within the month by WHO stage	WHO stage 1							
		WHO stage 2							
		WHO stage 3							
		WHO stage 4							
		Sub-total							
4.	Cumulative No. of persons started on ARVs at this facility at end of the month.								
5.	Total No. of patients currently on ARVs	Pregnant women							
		All others							
		Sub-total							
6.	No. of persons who are enrolled and eligible for ART but have not been started on ART								
7.	Post exposure prophylaxis(PEP)	Sexual assault							
		Occupational							
		All others							
		Sub-total							

DISTRICT NAME:_____ MONTH:_____ YEAR:_____ MOH 711_B

L: BLOOD SAFETY		Number
1.	Blood units collected from Regional Blood Transfusion Centers	
2.	Blood units collected from other sources Other than Regional Blood	
3.	Blood units screened at health facility	
4.	Blood units screened found HIV+	
5.	Blood Units transfused	

		NUMBER
6.	Blood units screened for hepatitis B	
7.	Blood units screened for hepatitis C	
8.	Blood units screened for syphilis	

Prepared By:		Designation:	
Date:		Signature:	

Figure 321: **Sample of MOH 711 Register Reporting Form**

Sample code

```
<?php

session_start();

require_once("lib.php");

require_once  'DB.php';

    $db = new DB();

date_default_timezone_set('Africa/Nairobi');

if(empty($_SESSION['userid'])){

redirect("modules/auth/users/login.php");

}
?>
<!DOCTYPE html>
<html>
<head>
<!-- Title -->
<title>DHIS2-PartnerHISs Data Exchange Bus</title>
<meta content="width=device-width, initial-scale=1" name="viewport"/>
<meta charset="UTF-8">
<meta name="description" content="Admin Dashboard Template" />
```

```
<meta name="keywords" content="admin,dashboard" />
<meta name="author" content="Steelcoders" />
<!-- Styles -->
<link href='http://fonts.googleapis.com/css?family=Open+Sans:400,300,600' rel='stylesheet' type='text/css'>
<link href="assets/myAssets/secondary/plugins/pace-master/themes/blue/pace-theme-flash.css" rel="stylesheet"/>
<link href="assets/myAssets/secondary/plugins/uniform/css/uniform.default.min.css" rel="stylesheet"/>
<link href="assets/myAssets/secondary/plugins/bootstrap/css/bootstrap.min.css" rel="stylesheet" type="text/css"/>
<link href="assets/myAssets/secondary/plugins/fontawesome/css/font-awesome.css" rel="stylesheet" type="text/css"/>
<link href="assets/myAssets/secondary/plugins/line-icons/simple-line-icons.css" rel="stylesheet" type="text/css"/>
<link href="assets/myAssets/secondary/plugins/offcanvasmenueffects/css/menu_cornerbox.css" rel="stylesheet" type="text/css"/>
<link href="assets/myAssets/secondary/plugins/waves/waves.min.css" rel="stylesheet" type="text/css"/>
<link href="assets/myAssets/secondary/plugins/switchery/switchery.min.css" rel="stylesheet" type="text/css"/>
rel="stylesheet" type="text/css"/>
<link href="assets/myAssets/secondary/plugins/slidepushmenus/css/component.css" rel="stylesheet" type="text/css"/>
<link href="assets/myAssets/secondary/plugins/x-editable/bootstrap3-editable/css/bootstrap-editable.css" rel="stylesheet" type="text/css">
<link href="assets/myAssets/secondary/plugins/x-editable/inputs-ext/typeaheadjs/lib/typeahead.js-bootstrap.css" rel="stylesheet" type="text/css">

<link href="assets/myAssets/secondary/plugins/x-editable/inputs-ext/address/address.css" rel="stylesheet" type="text/css">

<link href="assets/myAssets/secondary/plugins/select2/css/select2.min.css" rel="stylesheet" type="text/css">

<link href="assets/myAssets/secondary/plugins/bootstrap-datetimepicker/css/bootstrap-datetimepicker.min.css" rel="stylesheet" type="text/css">
<!-- Theme Styles -->
<link rel="stylesheet" href="assets/myAssets/css/style.css">
<link href="assets/myAssets/secondary/css/modern.min.css" rel="stylesheet" type="text/css"/>
<link href="assets/myAssets/secondary/css/themes/white.css" class="theme-color" rel="stylesheet" type="text/css"/>

<link href="assets/myAssets/secondary/css/custom.css" rel="stylesheet" type="text/css"/>
<script src="assets/myAssets/secondary/plugins/3d-bold-navigation/js/modernizr.js"></script>
```

```
<script src="assets/myAssets/secondary/plugins/offcanvasmenueffects/js/snap.svg-min.js"></script>
<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->
<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
<!--[if lt IE 9]>
<script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>

<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
<![endif]-->
<script type="application/javascript">
	$(function() {
		// a workaround for a flaw in the demo system (http://dev.jqueryui.com/ticket/4375), ignore!
		$( "#dialog:ui-dialog" ).dialog( "destroy" );
		$( "#dialog-form" ).dialog({
autoOpen: false,
modal: true,
		});
		$( "#nyef" )
			.button()
			.click(function() {

				$( "#dialog-form" ).dialog( "open" );
			});
	});

</script>
<script>
	$(function() {

		$( "#tabs" ).tabs();
	});
</script>
<script>

jQuery(document).ready(function () {

		$('input.date_input').datepicker({
changeYear: true,
yearRange: '1930:2100',

dateFormat: 'yy-mm-dd'

		});

	});
```

```
</script>
<script language="javascript" type="text/javascript">

functioncheckDate(field){

varallowBlank = true; varminYear = 1902; varmaxYear = 2099;
varerrorMsg = "";

// regular expression to match required date format
        //re = /^(\d{4})\/(\d{1,2})\/(\d{1,2})$/;

re = /^(\d{4})-(\d{1,2})-(\d{1,2})/;
if(field.value != ''){
if(regs = field.value.match(re)) {
if(regs[3] < 1 || regs[3] > 31) {
errorMsg = "Invalid value for day: " + regs[3];
            }
else if(regs[2] < 1 || regs[2] > 12) {
errorMsg = "Invalid value for month: " + regs[2];
            } else if(regs[1] <minYear || regs[1] >maxYear) {

errorMsg = "Invalid value for year: " + regs[1] + " - must be between " + minYear + " and "
+ maxYear;
            }
        }
else {

errorMsg = "Invalid date format: " + field.value;
        }
    }

else if(!allowBlank) {

errorMsg = "Empty date not allowed!";
    }

if(errorMsg != "") {

alert(errorMsg); field.focus();

return false;
    }
return true;
  }
</script>
<script type="text/javascript">
```

```
<!--
functionshowmenu(id){

var s = document.getElementById(id).style;

s.visibility='visible';
	}
	//-->
functiontimeOut(id){
setTimeout('hideShow('"+id+'')',3000)
	}

functionhideShow(id){

var s = document.getElementById(id).style;

s.visibility=s.visibility=='hidden'?'visible':'hidden';
	}
</script>

<script language="javascript" type="text/javascript">
varnewwindow;
functionpoptastic(url,h,w)
	{
varht=h;

varwd=w;

newwindow=window.open(url,'name','height='+ht+',width='+wd+',scrollbars=yes,left=250,to
p=80');
if (window.focus) {newwindow.focus()}
	}
functionplaceCursorOnPageLoad()
	{
if(document.stores)
showUser();
document.cashsales.itemname.focus();
	}
</script>
</head>

<?php

if (get_magic_quotes_gpc()){

   $_GET = array_map('stripslashes', $_GET);
   $_POST = array_map('stripslashes', $_POST);
```

```
   $_COOKIE = array_map('stripslashes',  $_COOKIE);
}
?>
<body style="background:#ccc;"  class="page-header-fixed">
<div class="overlay"></div><nav class="cbp-spmenucbp-spmenu-vertical cbp-spmenu-right" id="cbp-spmenu-s1"><h3><span class="pull-left">Chat</span><a href="javascript:void(0);" class="pull-right" id="closeRight"><i class="fafa-times"></i></a></h3><div class="slimscroll"></div></nav><nav class="cbp-spmenucbp-spmenu-vertical cbp-spmenu-right" id="cbp-spmenu-s2"><h3><span class="pull-left">Sandra Smith</span><a href="javascript:void(0);" class="pull-right" id="closeRight2"><i class="fafa-angle-right"></i></a></h3><div class="slimscroll chat"></div></nav>
<main class="page-content content-wrap">
<div class="navbar">

<div class="navbar-inner"><div class="sidebar-pusher"><a href="javascript:void(0);" class="waves-effect waves-button waves-classic push-sidebar"></a></div>
<div style="width:600px !important;;" class="logo-box">

<a href="index.php" class="logo-text"><span>DHIS2-PartnerHISs Data Exchange Bus</span></a>
</div>
<div class="topmenu-outer">
<div class="top-menu">
<ul class="navnavbar-navnavbar-right">
<li>
<a href="index.php" class="log-out waves-effect waves-button waves-classic">
<span class="glyphiconglyphicon-refresh "></span>

</a>
</li>
<li>
<a href="modules/auth/users/logout.php">

<span><i class="fafa-power-off m-r-xs"></i></span>
</a>
</li>
</ul><!--Nav -->
</div><!-- Top Menu -->
</div>
</div>
</div><!--Navbar -->
<div style="min-height:1088px !important; background: #ffffff !important;" class="page-inner">
<div id="main-wrapper" style="background: #ffffff !important;">
<div class="row">
<div class="col-md-1"></div>
<div class="col-md-4 login-info">
```

```
<div class="image-profile">
<img class="img-circle" style="-webkit-box-shadow:0 0 0 5px #fff;-moz-box-shadow:0 0 0 5px #fff;-o-box-shadow:0 0 0 5px #fff;box-shadow:0 0 0 5px #fff" src="assets/images/default-user.jpg">
<hr>
</div>
<div class="logged-in-user-info">
<p>Currently Logged in as:</p>
<span class="logged-in-name">

<?php echo $_SESSION['username']; ?>

<?

<button id="connect" onclick="window.location.href='modules/sys/config/connect.php'" type="button" class="btnbtn-primary btn-addon m-b-smbtn-lg">

<i class="fafa-line-chart"></i>DHIS2-PartnerHIS Data Exchange Bus
</button>
</div>
<div class="col-md-3">
<div class="panel panel-white">
<div class="panel-heading clearfix">

<h3 style="text-align: center !important;" class="panel-title">Account</h3>

</div>
</div><!-- Row -->
</div><!-- Main Wrapper -->

<p style="text-align: center; font-size:15px !important;" class="no-s">&copy; <?php echo(date('Y')) ?>  Licenced to:  <?php echo $_SESSION;?>

  J MwaiMsc. Computer Science UONBI. All Rights Reserved.
</p>
```

END