2005

Data mining framework

Hemambika Payyappillil
West Virginia University

DATA MINING FRAMEWORK

by

Hemambika Payyappillil

Thesis submitted to the College of Engineering and Mineral Resources at
West Virginia University in partial fulfillment of the requirements
for the degree of

Master of Science

in

Computer Science

Committee Members
Dr.V.Jagannathan, Ph.D., (Committee Chair)
Dr.Ramana Reddy, Ph.D.
Dr. Sumitra Reddy, Ph.D.

Lane Department of Computer Science and Electrical Engineering
Morgantown, West Virginia
2005

ABSTRACT

Data Mining Framework

Hemambika Payyappillil

The purpose of this document is building a framework for working with clinical data. Vast amounts of clinical records, stored in health repositories, contain information that can be used to improve the quality of Health Care. However, the information generated from these records depends vastly on the manner, in which the data is arranged. A number of factors need to be considered, before information can be extracted from the patient records. This document deals with the preparation of a framework for the data, before it can be mined.

One of the issues to deal with is information about the patient contained in the clinical records that can be used for identification purposes. A means to create anonymous records is discussed in this document. Once the records have been de-identified, they can be used for data mining. In addition to storing patient records, the document also discusses the possibility of 'abstracting' information from these documents and storing them in the repository. Information generated from the combination of patient records and abstracted information, could be used to improve the quality of Health Care.

This document also discusses the possibility of creating a means to query information from the data repository. A prototype application, which provides all these facilities in a form that can be accessed from any remote location, is discussed. In addition, the prospect of using Clinical Document Architecture format to store the clinical records is explored.

ACKNOWLEDGEMENT

I would like to express my gratitude to my advisor Dr.V.Jagannathan for his support and guidance through my thesis. I am also grateful to him for introducing me to new and interesting technologies in software development.

I am also grateful to my other committee members, Dr. Ramana Reddy and Dr.Sumitra Reddy for their valuable advice and support. I would also like to thank all the professors and staff of the Lane Department of Computer Science and Electrical Engineering.

I would also like to thank Medquist, Inc for their support in completing this project. Finally I would like to thank my family and friends for their constant help and support.

TABLE OF CONTENTS

1. Introduction 1

1.1 Problem Description 1
1.2 Objective 2
1.3 Overview 3

2. Background 5

2.1 Data Mining 5
2.1.1 Methodology of Data Mining 5
2.1.2 Analysis of the problem 6
2.1.3 Preparing Data 7
2.1.4 Data Modeling 8
2.1.4.1 Data Modeling Techniques 9
2.1.4.2 Selection of Data Modeling technique 10
2.1.5 Data Mining Applications 11
2.2 Data Mining and Health Care 11
2.2.1 Mining the data 12
2.2.2 Applications 13
2.3 Quality Measures 13

3. Technology 16

3.1 Clinical Document Architecture 16
3.1.1 CDA Release One 17
3.1.2 CDA Release Two 21
3.1.3 Differences between CDA Release One and CDA Release Two 25
3.2 XML Databases 25
3.2.1 XML Enabled Databases 27
3.2.2 Native XML Databases 28
3.3 XQuery 29
3.3.1 Applications 29
3.3.2 Requirements 30
3.3.3 XQuery Expressions 30

4. System Overview 33

4.1 Goals 33
4.2 System Architecture 34
4.3 System Evolution 36
4.4 Constraints 36
4.5 Benefits 37

5. Design 38

5.1 Description 38
5.2 UML Diagrams 39
5.2.1 Sequence Diagrams 39

5.2.2 Class Diagram 42
5.3 Classes and Functions 43
5.3.1 Anonymous class 43
5.3.2 Abstraction class 44
5.3.3 Query class 45
5.3.4 Mapping class 45
5.3.5 TigerLogic class 45

6. Implementation 47

6.1 Database Management System 47
6.1.1 SQL Server 2000 47
6.1.2 Oracle 8i 48
6.1.3 TigerLogic 2.0 48
6.1.4 Chosen Database Management System 50
6.2 Application Development Environment 50
6.3 Programming languages 52
6.4 Web Interfaces 53

7. Analysis 60

7.1 Testing 60
7.1.1 Anonymous module 60
7.1.2 Abstract module 61
7.1.3 Query Module 61
7.2 Improvements 62
7.3 Conclusion 62

References 64

LIST OF FIGURES

Sample CDA Release One Document Header 20
Sample CDA Release One Document Body 20
Sample CDA Release Two Document Header 24
Sample CDA Release Two Document Body 24
System Architecture 34
Sequence diagram for Anonymous Data Module 39
Sequence diagram for Data Abstraction Module 40
Sequence diagram for Data Query Module 41
Class diagram 42
View of a Clinical Document Architecture 1.0 document in TigerLogic 49
Web Interface-Main Page 53
Web Interface-Data Abstraction 54-56
Web Interface-Data Query 57-58
Web Interface-Anonymous Data 59

LIST OF TABLES

Hospital Quality Alliance Starter Set 14
Differences between CDA Release 1.0 and 2.0 Documents 25

INTRODUCTION

Data Mining is the science of finding patterns in huge reserves of data, in order to generate useful information from it. Data Mining has potential applications in several fields, not the least of which is Health Care. The myriad possibilities of improvement in Health Care through Data Mining only further justify the need to apply data mining principles to clinical data. However, prior to applying data mining techniques to garner information from data, the data has to be 'prepared' to ensure the veracity of the information obtained. 'Preparing' the data involves removal of incorrect information or 'noise' from the data and ensuring that the data mining principles are applied on real data. This document gives a detailed description of the purpose, design and implementation of the Data Mining Framework. The primary purpose of the Data Mining Framework is to help determine trends in patient records to improve Health Care.

1.1 Problem Description

The term 'Patient Encounter', for the remainder of this document will refer to the time frame, from when the patient is admitted to the hospital, to the time he is discharged from the hospital. The Medical records of a patient consist of a large number of reports, generated for all medical tests conducted, during each encounter. An effective means of storage of these encounter reports should be devised, so that relevant information can be extracted as and when required. For example, it might be necessary to ascertain if a certain medicine was administered to the patient on arrival at the hospital and the effect of that medicine on the health of the patient. Retrieval of this information should be trivial.

A standard notation for representation of the data in the encounter reports must be followed. Further the representation chosen should be language and platform independent so that it allows easy transfer of data over a network. In addition, the data representation should be amenable to detecting trends in the clinical data. For example, it might be necessary to determine the effect of a certain medicine to combat a disease. This can be learnt from studying a number of encounter reports.

The encounter reports might consist of some information, which might encroach on the privacy of the patient, such as the patient's name and address. Before any Data Mining techniques can be applied to these encounter reports, we must ensure that patient privacy is protected. Regulations such as the Health Insurance Portability and Accountability Act (HIPAA)

require that the patient confidentiality be preserved. A means to de-sensitize the information needs to be provided.

In addition to anonymous data and effective storage and representation of data, the Framework should provide a means to 'abstract' data. 'Abstraction' is a term used in Information Extraction and refers to the science of analyzing a document to extract relevant information from it. The National Voluntary Hospital Reporting Initiative launched by the American Hospital Association (AHA), the Federation of American Hospitals (FAH), and the Association of American Medical Colleges (AAMC) requires that ten quality measures, be reported from the encounter reports of all patients. The quality measures, which would need to be reported, as part of this initiative, are explained in the following chapter. The Data Mining Framework should provide a means for manual abstraction of these quality measures form the encounter reports.

The extracted quality measures are stored in the database. Data mining techniques could be applied to the abstracted data to improve health care. For example, the number of heart patients who had been administered aspirin on arrival could be determined from the abstracted data. This could be used to learn the effects of administering aspirin to heart patients on arrival at the hospital and thereby lead to an overall improvement in the quality of health care.

The Data Mining Framework should also provide means to query the database to get relevant information from the database. Some possible applications for this would be to predict the onslaught of an epidemic or to study the prevalence of a particular disease in a specified age group or sect of people. For example a query could be made to find the number of people suffering from typhoid in a particular region, which could then used to predict an epidemic. Similarly queries could be run against the database to find the number of patients below the age of forty, who suffer from diabetes.

1.2 Objective

The goal of this thesis is to create an application called Data Mining Framework, which can provide an infrastructure for Data Mining purposes. XML is the chosen representation for the data in the encounter reports. The standard that will be followed to represent the Health Care Data is the Clinical Document Architecture (CDA). The recognized version of CDA documents used at the time of this implementation was CDA Release One. The newer version viz. CDA Release Two has different element tags from CDA Release One. In addition to being structurally

different from CDA Release One, CDA Release Two allows encoding of values. The Data Mining Framework designed should be flexible to work with both versions of CDA documents. The database managed by this application will be made public via the world-wide-web through user-friendly interfaces. The interfaces would allow query and abstraction of the data. Data Scrubbing or creating anonymous data would also be possible. The objectives can be summarized as follows:

- Create a data-mining infrastructure that can be accessed from any authorized machine via the Internet.
- Ensure that the system is flexible to work with all versions of Clinical Document Architecture (CDA) formats
- Secure privacy of the patient by masking sensitive information such as names and addresses.
- Abstract the quality measures stipulated in the National Voluntary Hospital Reporting Initiative from all the encounters of a patient
- Allow queries to be run against the database, which would help improve Health Care Quality
- Use a Database Management System that natively supports XML documents for effective storage and retrieval of data.
- Use an object oriented approach when designing the system, to ensure that the system is scalable

1.3 Overview

The primary purpose of this document is to explain the purpose, design and implementation of the Data Mining Framework. Chapter 2 gives an over-view of the literature review conducted for this system. Principles in Data Mining and its applications in Health Care are discussed. The starter set of quality measures, which are abstracted from the clinical documents are also listed. Chapter 3 discusses the technology used in developing the Data Mining Framework. A description of the support for XML documents found in current Database Management Systems is given. A study of the Clinical Document Architecture (CDA) format is also made. A brief introduction to XQuery is also included. Chapter 4 gives a functional overview of the system. The system architecture and functions of the Data Mining Framework

are discussed. Chapter 5 describes the design of the system. Detailed UML diagrams such as sequence and class diagrams depict the functioning of the system. Chapter 6 describes the tools used in implementing the Data Mining Framework. Briefly, the Database Management System used, the application development environment and programming language used are discussed. Chapter 7 discusses future improvements, which could be made to the Data Mining Framework. The test cases used to analyze the performance of the application are also discussed.

2.BACKGROUND

This chapter discusses Data Mining techniques and principles in detail. Applications of data mining to Health Care are also discussed.

2.1 Data Mining

Data Mining is the science of extracting useful information from large amounts of data. It finds patterns in the data and can make predictions about trends in data. This is particularly useful for businesses to make informed decisions about future prospects. For example, if the data consisted of sales information, collected over several years, data mining techniques could be used to predict trends in consumer demand. This can be used to improve sales performance.

The core components of Data Mining are the vast collections of un-mined data, powerful multiprocessors and data mining algorithms. The importance of the collection of un-mined data cannot be undermined. The data has to be comprehensive, so that trends predicted can be accurate. Moreover, care should be taken to clean the data. Data Cleaning is removal of repetitive, erroneous and misleading data, which can result in errors in trend predictions. Powerful multiprocessors are always an aid to data mining. They can produce fast and accurate outputs for the data mining algorithms, applied on the data.

Data Mining can automate the process of finding trends and patterns in databases. Large Databases can be analyzed quickly and effectively, using high performance systems and time effective algorithms. As the speed of processing improves, the size of the database can be increased. Also, as the size of the database increases, the accuracy of the predictions also improves. Co-relation among data can be identified. For example, patterns in the sales data of a company can predict a co-relation between seemingly unrelated products. These products can then be sold together, thereby improving sales.

2.1.1 Methodology of Data Mining

Data mining works on the principle of modeling. The idea here is that, the solution for a given problem is determined, from an existing solution of a known problem. The primary requirement to achieve this is to collect a huge set of data for an existing solution. This data could be fed to large and powerful processing machines, which consist of data mining tools. These tools work on the data and identify patterns and trends for that model of data. Once the

trends have been identified, this model of data could be applied to situations, where, the solution is not known. The primary steps involved in the process of data mining can be summarized as follows:

- **Analyzing the problem**: The primary step in the process of data mining is the analysis of the problem. It has to be determined, if there would be sufficient data available to be mined. The data mining techniques to be used are determined.
- **Preparing the data**: Preparing the data is one of the most important facets of data mining. The data has to be cleaned to remove erroneous and repetitive information. The accuracy of the predictions is to a large extent, dependant on the accuracy of the data. Further, the data has to be aggregated and converted to a form that is required by the data mining algorithms.
- **Modeling:** Data Mining model is prepared, which can be used by the Data Mining tools to identify trends and patterns in the existing data. Powerful data mining algorithms are used to achieve this end. The determined patterns are then used to make predictions about the data and co-relation between the data. The kind of knowledge discovery task i.e. prediction or description, to be performed is determined.
- **Deployment of Results:** A summarization of the results obtained is performed. A strategy for the maintenance of the created model is determined

2.1.2 Analysis of the problem

The primary step in the data mining process is to understand the problem at hand. The problem might consist of several related mini-problems. The objective would be to identify the constraints of the problems and the factors, which have a bearing on the outcome. The next important step is to identify the criterion for success. This can be subjective or objective. In the case of the former, domain experts, who have knowledge of the co-relation between domain variables, can judge the percentage of success. In the case of the latter, the success is determined by the amelioration in the problem.

The next step in the analysis of the problem would be to determine the extent of expertise available to solve the problem. The terminology for the problem should also be defined. This would improve the co-ordination between domain experts and data mining experts. Further, the potential cost of the data-mining project should be determined. If the benefits that would be

derived as a result of implementing this project do not over ride the potential cost, then the proposed project could be shelved.

The data, which needs to be mined, has to be collected. A document has to be maintained, which lists out the data, which has been collected, the locations from which it has been collected and the locations in which it is being held. The problems encountered, while collecting the data and the subsequent solutions should also be mentioned. A description of the data attributes and their relationship should also be documented. Validation of the data could also be performed. Missing values in the attributes are identified and dealt with. Erroneous data should also be identified and the possible means to overcome it need to be specified. The proposed data mining techniques, tools and algorithms to be used should also be finalized.

2.1.3 Preparing Data

Preparing the data, prior to applying data mining techniques, is a critical step of data mining. It primarily consists of four steps. The first step is the selection of data. This is following by data cleaning, forming the new data and finally formatting the data.

Data has to be selected, prior to applying data mining principles. The data is chosen based on its completeness and correctness. Constraints on the data, such as the data type, could also be factors in the selection of the data.

Data Cleaning is the most important and consequently the most time taking process in Data Preparation. The importance of cleaning the data cannot be undermined. The accuracy of the data mining predictions, depend on a large scale on the accuracy of the data. Data Cleaning involves removal of erroneous or incomplete data. The following methods could be followed for cleaning the data:

- The data could be normalized. For example the data could be scaled to a pre-defined range. This is useful to fill in gaps or incomplete data.
- Any numeric attributes, if present, could be discretized. This is extremely useful for certain data mining techniques.
- Missing attribute values could be filled. Although, there are several solutions to filling the missing data, it is advisable to perform the modeling of the data with and without these attributes. This identifies the importance of those attributes. Some of the possible solutions to filling the missing attribute values include replacing the missing values with

a global constant. Alternately, the missing attribute values could be filled with the feature or class mean. However the problem with these solutions is that it could lead to a biased prediction, since the substituted value is not the correct value. Other solutions include, deleting the attributes containing missing values, from the set. Yet other solutions include predicting the missing values, using data mining tools.

- Reduction of data could be performed. This is performed primarily, if the run time to obtain the solution is too long or if that data set is too large. The easiest form of data reduction is to examine candidate attributes for their influence on the outcome. If they do not influence the outcome to a large degree, they can be discarded. The candidate attributes could be selected from means and variances, from principal component analysis or from linear transformations.

Creation of new data can be done using one of several means. Some of the possible ways to reconstruct data are to derive new attributes from two or more existing attributes. Alternatively data could be merged or aggregated to form new data.

Formatting the data is an important step of Data Preparation. Some of the formatting steps include re-ordering the attributes, such as placing the target attributes at the beginning or at the end of the data set. This could also include creating a random order for the values in the data set. Further all special characters such as commas, tabs and other trimming characters should be replaces with the allowed set of special characters for that data set. Conversion of the data set into a standard form is also part of the formatting process.

2.1.4 Data Modeling

The data-mining tool, based on the type of task that has to be performed, creates the data model. The modeling phase is an important aspect of the data-mining process. There are a vast number of data-mining techniques that can be applied to the data model. The data modeling technique, which is applied depends on the type of the model. The trends and patterns that are found in the data are based on the data mining technique applied. These patterns can be fed to decision support systems, which can make informed predictions on the data. Some of the more popular data modeling techniques are discussed in the following section.

2.1.4.1 Data Modeling Techniques

Some of the popular data mining techniques used are as follows:

a) Decision Trees

Decision tree is an attribute classifier, which takes the form of a tree. It can consist of leaf nodes or internal nodes. The former signify the value of an attribute while the latter are decision nodes. These nodes, as the name signifies, specify a test that can be performed on the attribute, so that the attributes can be classified as sub-trees. Decision trees algorithms basically use the divide-and-conquer approach to classify. They thus work in a top-down manner, using an attribute at each level split on, that best separates the classes. Decision trees signify rules and are easily interpretable by humans. Some of the more popular decision tree methods are Classification and Regression trees

There are several algorithms for building decision tress, such as the popular C 4.5 algorithm. The most important factor of an algorithm used for building a decision tree is selecting the attribute, which forms the decision node. There are several issues to be considered, when creating a decision tree. These include determining the depth of the tree, working with continuous attributes and determining the appropriate means of selecting the attribute. Additional issues include working with missing values, over-fitting and attributes that have variable costs.

b) Rule Induction Methods

In Rule Induction methods, rules are created from the data based on statistical values. In this approach, all values in a class are considered. At each stage a rule is generated for these values. Testing the rule under construction with new values creates a rule with maximum accuracy. The value is chosen such that it maximizes the probability of the desired classification. One of the popular rule induction methods is the Prism method.

c) Association Rules

Association rules signify relationship between attributes. They depict how the attributes are related to each other, regardless of the class value. The value of the rule is determined by its *confidence.* The confidence of a rule is the ratio of the number of times the rule is supported to the number of times the conditional part of the rule is supported. The higher the confidence, the better is the value of the rule. The most important factor of

an association rule is the choice of attributes that become part of the rule. Although more than two attributes could be used in the conditional part of the rule, care should be taken to ensure that the combinations of attributes does not become too large. The primary application for association rules is in market analysis because of the clarity of its results.

d) Clustering Techniques

Clustering techniques try to find patterns in the data as a whole. They function by dividing the data into clusters. All data in a cluster is homogenous and is different from data in other clusters. The significant difference between this technique and the previous techniques is the absence of target variables. Thus no differentiation is made between dependent and independent variables.

e) Neural Networks

Neural networks, as the name signifies is the inter-connection of elements called neurons. The input to a neuron can be any number of values. The neuron computes a single output for these input values using a function. Usually the input and output values have continuous values. In complex neural networks, the output of some neurons can be connected to the input of other neurons. Neural Networks out perform other techniques, when the domain is non-linear or noisy. However, Neural Networks are slow learning processes.

2.1.4.2 Selection of Data Modeling technique

It is difficult to determine the best modeling technique. Each technique has its own advantages and limitations. The one that is best suitable to the problem at hand should be chosen. One method to determine the most suitable modeling technique is by trial and error. Alternatively, the technique chosen should be based on the kind of knowledge discovery task that needs to be performed. If the task to be performed is Prediction then, Neural Networks or Regression trees could be used. If the task to be performed is Classification, then Decision trees and Rule-Induction methods are preferred.

2.1.5 Data Mining Applications

Data Mining can find applications in several fields. Some of the possible applications are:

- ***Banking**:* Can be used to create risk assessment models in the finance industry.
- ***Pharmaceutical Industry:*** Can be used for building visualization tools for genomics and discovery of drugs.
- ***E-Commerce:*** Can be used to deliver marketing messages, which are personalized, to the customer. Can target and interact with the customer.
- ***Health Care:*** Can be used to predict onslaught of epidemics. Can be used to learn the effects of a certain disease on a particular age group or race. Can be used to improve quality of Health Care.
- ***Marketing:*** Can be used to understand the co-relation between products. For example, if the sale of a certain product depends on the sale of another product, this can be identified and both the products can be marketed together.

2.2 Data Mining and Health Care

The influence of data mining on the quality of Health Care cannot be understated. All Health Care organizations retain detailed and comprehensive records of patient data. Trends and patterns identified in these records can positively impact the quality of Health Care. The huge amounts of patient data, makes identification of these trends an arduous task. However data mining applications, built for this purpose, can make this very simple and produce efficient results.

There have been several cases, where application of data mining techniques, have helped resolve a problem in the health industry. For instance, data mining on pneumonia patient records in a hospital, showed that patients who were administered medication immediately on arrival responded better than patients who were not administered medication on arrival. In order to arrive at this conclusion the data mining application, used several inputs, such as the tests and other information of the patients who showed better medication results. Various relations were drawn between the inputs. One of these was the relation between the results and the time taken to administer medication after arrival. It was found that, shorter the time, better the result.

There were several other key issues that were addressed at this time. The data mining tests proved that several tests, which were largely extraneous, were conducted on the patients.

These led to a delay in the administration of medication and thereby affected the recovery of the patient. To overcome this, a standardized plan was created to treat pneumonia patients. The identification of these associations between inputs and finding the resultant best outcome was possible only because of data mining techniques.

2.2.1 Mining the Data

Health care organizations store huge amounts of data in the form of patient databases. Trends in these databases can be identified using data mining practices, which sort and model the data in order to arrive at a conclusion. The data mining applications present the data in the form of data marts. This allows end users to choose the specific sets of data, which they want to be analyzed. The data in these data marts can then be presented using a graphical user interface, arranging the data into columns and rows.

In the Health care industry, however, the lack of standard clinical vocabulary has hindered the process of data mining to a certain extent. For example a simple term such as 'hypertension' can be expressed in various ways in health care. This could lead to unnecessary problems, during the process of data mining. The increase in the use of standardized terms will reduce the percentage of errors in the data mining process.

Cleaning the data before it can be mined is also an important step in the data mining process. In many Health care organizations, the mode of preparing patient reports can lead to a good deal of confusion. For instance, in a certain hospital, a report was prepared, before and after a patient went in for an X-ray check. This could be construed as two different reports, when analyzing the data and produce erroneous results. Further in certain organizations, in order to reduce the number of reports, a patients' record contains only the name of the attending physician and not the names of other physicians consulted or tests performed at a later stage, leading to erroneous predictions.

The data mining effort thus requires the wholehearted participation of all health care personal to produce comprehensive and correct reports, which can be mined. Further, the number of input variables for the data mining application has to be determined correctly. The number of inputs should not be so large, that it produces confusing results. The variables that need to be studied have to be pre-determined, so that the meaningful associations are produced at the end of the data mining process. It is a good idea to limit the number of variables. However they should

not be limited to such an extent, that they produce biased results. Co-operation between the physicians and analysts is also recommended, since some of the results might be more easily understood by the health care personal.

2.2.2 Applications

The following are some of the applications of data mining in the Health care industry:

- Avoid member attrition in the Health Insurance industry. Further patterns in the existing members can be used to increase the member population
- Fraudulent claims of Health Insurance can be determined. Using techniques such as classification and regression, members with questionable activities can be identified.
- Members who are more receptive to additional insurance coverage can be identified.
- Recruiting and retaining health care staff
- Monitor patient results and identify significant variations in health.
- Management of diseases and prediction of epidemics

2.3 Quality Measures

The Centers for Medicare & Medicaid Services (CMS), in an effort to improve the quality of Health Care, has initiated the Hospital Quality Alliance (HQA). HQA is a National Voluntary Initiative by all Hospitals to report health related data, which can be used to improve Quality of Health Care. HQA is a combined effort of the American Hospital Association, the Federation of American Hospitals, the American Association of Medical Colleges, the Joint Commission on Accreditation of Healthcare Organizations, National Quality Forum, the American Nurses Association, the American Medical Association, the AFL-CIO, AARP, the Consumer-Purchaser Disclosure Project, the National Association of Children's Hospitals and Related Organizations, CMS and AHRQ.

The Data Mining Framework provides a means to abstract the starter set of quality measures, identified by the Hospital Quality Alliance. The abstracted measures are stored in the Database. These abstracted measures could be reported using Reporting tools or Data mining techniques could be applied to learn from this data.

The quality measures that comprise the starter set of the measures of the Hospital Quality Alliance, concentrate mainly on three areas viz. Heart Attack, Heart Failure and Pneumonia. The

effect of certain pre-determined factors to help patients suffering from the above mentioned conditions are reported as quality measures. The starter set of quality measures of the Hospital Quality Alliance, is listed below:

Hospital Quality Alliance Starter set

Performance Measures	**Measure Description – for additional information including inclusions and exclusions click on the Performance Measure**
AMI - Aspirin at Arrival	Acute myocardial infarction (AMI) patients without aspirin contraindications who received aspirin within 24 hours before or after hospital arrival.
AMI - Aspirin Prescribed at Discharge	Acute myocardial infarction (AMI) patients without aspirin contraindications who are prescribed aspirin at hospital discharge.
AMI – ACEI for LVSD	Acute myocardial infarction (AMI) patients with left ventricular systolic dysfunction (LVSD) and without angiotensin converting enzyme inhibitor (ACEI) contraindications who are prescribed an ACEI at hospital discharge.
AMI - Beta Blocker at Arrival	Acute myocardial infarction (AMI) patients without beta blocker contraindications who received a beta blocker within 24 hours after hospital arrival.
AMI - Beta Blocker at Discharge	Acute myocardial infarction (AMI) patients without beta blocker contraindications who are prescribed a beta blocker at hospital discharge.
HF-LVF Assessment	Heart failure patients with documentation in the hospital record that left ventricular function (LVF) were assessed before arrival, during hospitalization, or planned for after discharge.
HF-ACEI for LVSD	Heart failure patients with left ventricular systolic dysfunction (LVSD) and without angiotensin converting enzyme inhibitor (ACEI) contraindications who are prescribed an ACEI at hospital discharge.

PNE-Initial Antibiotic Timing	Pneumonia patients who receive their first dose of antibiotics within 4 hours after arrival at the hospital.
PNE- Pneumococcal Vaccination	Pneumonia patients' age 65 and older that were screened for pneumococcal vaccine status and were administered the vaccine prior to discharge, if indicated.
PNE-Oxygenation Assessment	Pneumonia patients who had an assessment of arterial oxygenation by arterial blood gas measurement or pulse oximetry within 24 hours prior to or after arrival at the hospital.

Source: Official website of Hospital Quality Alliance

3.TECHNOLOGY

This chapter discusses in detail the technologies used to implement the Data Mining Framework

3.1 Clinical Document Architecture

The encounter reports used in the Data Mining Framework are expressed in Clinical Document Architecture (CDA) format. CDA is a recognized standard for representing clinical data. It is a Health Level 7 (HL7) standard that allows exchange of clinical data, by encoding data in Extensible Markup Language (XML). It specifies the structure and description of clinical documents. A CDA document can include text, images and other multimedia content. A CDA document consists of a header and a body. The header of the document sets the context of the document. In general it gives information about the authentication, patient, encounter and provider. The body of the CDA document consists of the medical report of the patient. This can be represented in structured form or unstructured form (BLOB of data).

The benefits of using CDA documents to represent patient information are many. Not the least among them, being the ability to have a cost effective implementation across multiple systems. The encoding in XML ensures compatibility across different platforms and languages. It can be legally authenticated. Unlike messages, it remains unaltered for the time defined time period. However, confidentiality and security of the CDA documents depend on the application systems, responsible for sending and receiving it. In addition to being machine readable, it is human readable for different technical levels of user sophistication. Further CDA documents are independent of the method of storage. Thus a CDA document can be stored as an independent file or as part of a native XML database or within a database management system.

CDA documents, represented in XML need to be validated and this is done against CDA Schema. A document that complies with this schema is said to be a conformant CDA document. CDA documents are extremely flexible. They can be extended to add meaning for local requirements. This means it is possible to add XML elements and attributes to the original XML Schema. However, it must be ensured that the inclusion of these new items does not change the meaning of the standard data items. Extensions can also be performed on the HL7 Namespace. Adding the “HL7extension” attribute to the document can do this. The Data Mining Framework is designed to be flexible. It is designed to work with all valid releases of CDA.

3.1.1 CDA Release One

CDA documents consist of a set of hierarchical XML schemas or document type definitions called levels that are semantically related. CDA Level One consists of a very detailed document header. The body of the document is structurally defined. It can be used to represent largely descriptive clinical reports. It does not support the complex HL7 version 3 messaging semantics. CDA Level Two is the layer above the CDA Level One specification. It specifies rules or constraints for the structure of the document. In other words, it can be used to specify mandatory and optional sections of the document. For instance, a constraint may specify that all documents of type “physical” must contain a mandatory “physical examination” section. However the implementation of this level would require the constraints to be defined by a large body of professional societies. Hence the CDA standard only defines the first Level. CDA Level three will make the CDA document more compliant to the HL7 Reference Information Model (RIM). This would allow clinical information to be depicted in HL7 version 3 messages. This would allow detailed descriptions of clinical information within the document and also allow extraction of relevant information.

The root element for a CDA Level one document is <levelone>. As stated formerly, a CDA document consists of a header and a body. The root element for the header is <clinical_document_header>. The header is responsible for exchange of clinical data between applications and also for providing a context to the entire document. The header contains information about the document, such as identification information and its relation to other documents. For example, the headers of a set of documents could be used to identify that they are all different versions of the same report. In addition it contains information about the patient encounter, such as the time of encounter. Further, it describes the ‘service actors’ and ‘service targets’. The service actors are the people responsible for creating and authenticating the document, such as the transcriptionists and health care personnel, who provided the services for the encounter being documented. The service targets are the people to whom the services are being provided, such as the patients and their family members.

Some of the element tags used in the header that describe the context of the document are as follows:

- **< id >** This element tag is used to uniquely identify the document. Any document, regardless of it being an original document or a replacement document, will get a unique value for this element tag.
- **<set_id>** The value in this tag is used to indicate, if the document is unique or if it is a document which replaces an already existing document. Unique documents have unique values, whereas replacement documents have the same value as the documents they are replacing.
- **<version_nbr>** This indicates the version number of the document. The version number is incremented by one, each time a document replaces an existing document. Thus the version number of the replacement document will be one greater than the version number of the parent (original) document.
- **<document_relationship>** This indicates the relationship of the document to other documents, such as if it is a replacement or appendage to an already existing document.
- **<fulfills_order>** A document could be associated with an order. The value in this tag associates the document uniquely to the orders that it fulfills.
- **<document_type_cd>** The document type code is used to classify the document. The values for this tag are obtained from the Logical Observation Identifiers, Names and Codes (LOINC)
- **<origination_dttm>** The value in this tag gives the time the document was created.
- **<encounter_tmr>** This indicates the time interval of the patient encounter.
- **<participation_tmr>** This tag represents the time interval of participation for the service actors and service targets.
- **<confidentiality_cd>** This indicates the status of the confidentiality for the document. The confidentiality can be applied to the whole or parts of the document.
- **<practice_setting_cd>** This is an indication of the medical setting such as a cardiology clinic.
- **<authenticator>** This element tag contains information about the personal who have validated the document.
- **<legal_authenticator>** This tag exists if the document has been legally validated by the person legally responsible for the document.

- **<signature_cd>** This contains the documentation of the signature of the person who has authenticated the document.
- **<originator>** This element tag gives information about the author of the document.
- **<originating_device>** This contains a value, if the author of the document is not a human source, but a machine.
- **<intended_recipient>** Copies of the CDA document can be sent to a number of people. This element tag indicates the recipients of the document.
- **<originating_organization>** This element tag provided information about the organization responsible for maintaining the document.
- **<transcriptionist>** This element tag provides details about the transcriptionist of the document.
- **<provider>** This lists the Health Care providers concerned with this document.
- **<patient>** This element tag is concerned with providing information about the patient, whose reports is being documented.
- **<birth_dttm>** This indicates the birth date of the patient
- **<administrative_gender_cd>** This element which stands for administrative gender code, indicates the gender of the patient
- **<service_target>** This contains information about the beneficiaries of the patient, such as family members

The root element for the body of the document is <body>. The body of the documents contains the actual report generated for the patient by the Health Care Providers. The element tags in the body of the document are as follows:

- **<coded_entry>** This allows the usage of coding schemas recognized by HL7 in CDA documents. This facilitates indexing the documents for search and retrieval. It also allows insertion of codes recognized locally.
- **<content>** This allows textual description of the disease, it's causes, symptoms and the necessary measures taken to overcome it. It can be recursively nested.
- **<link>** This is a reference mechanism and is analogous to a HTML anchor tag
- **<observation_media>** This is a reference to media, that is part of the CDA document.

```
<?xml version="1.0" ?>
<!DOCTYPE levelone (View Source for full doctype...)>
- <levelone>
  - <clinical_document_header HL7-NAME="document_service_as_clinical_document_header" T="service" RIM-VERSION="0.98">
      <id EX="--" T="II" EX-T="ST" EX-HL7_NAME="extension" RT-T="OID" RT-HL7_NAME="root" AAN-T="ST" AAN-
        HL7_NAME="assigningAuthorityName" VT-T="IVL_TS" VT-HL7_NAME="validTime" PROB-T="REAL" PROB-
        HL7_NAME="probability" HL7-NAME="id" />
      <set_id EX="MQ21493454" T="II" EX-T="ST" EX-HL7_NAME="extension" RT-T="OID" RT-HL7_NAME="root" AAN-T="ST" AAN-
        HL7_NAME="assigningAuthorityName" VT-T="IVL_TS" VT-HL7_NAME="validTime" PROB-T="REAL" PROB-
        HL7_NAME="probability" HL7-NAME="set_id" />
      <version_nbr T="INT" V-T="ST" V-HL7_NAME="value" VT-T="IVL_TS" VT-HL7_NAME="validTime" PROB-T="REAL" PROB-
        HL7_NAME="probability" HL7-NAME="version_nbr" />
      <document_type_cd V="73" SN="Medquest|Net" DN="Transcribed report" T="CE" V-T="ST" V-HL7_NAME="code" DN-T="ST" DN-
        HL7_NAME="displayName" S-T="OID" S-HL7_NAME="codeSystem" SN-T="ST" SN-HL7_NAME="codeSystemName" SV-T="OID"
        SV-HL7_NAME="codeSystemVersion" ORIGTXT-T="ST" ORIGTXT-HL7_NAME="originalText" VT-T="IVL_TS" VT-
        HL7_NAME="validTime" PROB-T="REAL" PROB-HL7_NAME="probability" HL7-NAME="service_cd" />
      <origination_dttm T="TS" V-T="ST" V-HL7_NAME="value" VT-T="IVL_TS" VT-HL7_NAME="validTime" PROB-T="REAL" PROB-
        HL7_NAME="probability" HL7-NAME="origination_dttm" />
    - <fulfills_order HL7-NAME="is_source_for_service_relationship" T="service_relationship">
        <fulfills_order.type_cd V="FLFS" T="CS" V-T="ST" V-HL7_NAME="code" DN-T="ST" DN-HL7_NAME="displayName" HL7-
          NAME="type_cd" />
      - <order HL7-NAME="has_target_service" T="service">
          <id T="II" EX-T="ST" EX-HL7_NAME="extension" RT-T="OID" RT-HL7_NAME="root" AAN-T="ST" AAN-
            HL7_NAME="assigningAuthorityName" VT-T="IVL_TS" VT-HL7_NAME="validTime" PROB-T="REAL" PROB-
            HL7_NAME="probability" HL7-NAME="id" />
        </order>
      </fulfills_order>
    - <patient_encounter HL7-NAME="is_assigned_to_patient_encounter" T="patient_encounter">
        <id EX="00012248019" T="II" EX-T="ST" EX-HL7_NAME="extension" RT-T="OID" RT-HL7_NAME="root" AAN-T="ST" AAN-
          HL7_NAME="assigningAuthorityName" VT-T="IVL_TS" VT-HL7_NAME="validTime" PROB-T="REAL" PROB-
          HL7_NAME="probability" HL7-NAME="id" />
        <encounter_tmr V="09/23/2003" T="IVL_TS" V-T="TS" V-HL7_NAME="value" VT-T="IVL_TS" VT-HL7_NAME="validTime"
          PROB-T="REAL" PROB-HL7_NAME="probability" HL7-NAME="encounter_tmr" />
      </patient_encounter>
    - <originator HL7-NAME="has_service_actor" T="service_actor">
        <originator.type_cd V="AUT" T="CS" V-T="ST" V-HL7_NAME="code" DN-T="ST" DN-HL7_NAME="displayName" HL7-
          NAME="type_cd" />
```

Sample CDA Release One Document Header

```
    </section>
  - <section>
      <caption>INJURIES</caption>
    - <paragraph>
        <content>Fracture of the nose, left elbow, forearm, concussion times two, and sports injuries. He has torn his right
          knee ligaments. Most recent injury was being tossed from a dirt bike and the bike did fall on him, dislocating the
          left elbow.</content>
      </paragraph>
    </section>
  - <section>
      <caption>MEDICAL PROBLEMS</caption>
    - <paragraph>
        <content>None major. He states he has had some problems in the past with cervical neuropathy, probably viral,
          involving the left side.</content>
      </paragraph>
    </section>
  - <section>
      <caption>PAST SURGICAL HISTORY</caption>
    - <paragraph>
        <content>An appendectomy, a left neck lymph node dissection, a colonoscopy times two.</content>
      </paragraph>
    </section>
  - <section>
      <caption>HABITS</caption>
    - <paragraph>
        <content>He smokes since the age of 19, and he has tried to quit but has not had any success at that. Alcohol:
          Probably more than one a day.</content>
      </paragraph>
    </section>
  - <section>
      <caption>FAMILY HISTORY</caption>
    - <paragraph>
        <content>His father is deceased. He did have uncontrolled diabetes and severe peripheral arteriosclerosis. Mother
          died at the age of 66 of cancer of the lung. Siblings: One sister living with fibrocystic disease of the left
          breast.</content>
      </paragraph>
    </section>
```

Sample CDA Release One Document Body

3.1.2 CDA Release Two

CDA Release Two documents, like the previous Release, contain notion of levels. However, unlike CDA Release One, it does not contain a hierarchical set of XML Schemas. It consists of a single XML Schema and a hierarchical set of HL7 templates. The two main categories of HL7 templates are section-level templates and entry-level templates. The former is concerned with imposing constraints at the section level, depending on the type of the document, while the latter imposes constraints within the document sections. The root element of the document is the <ClinicalDocument>. It contains the header and body elements. The body elements are contained in the <StructuredBody> element and can have a nested structure, while the header elements are contained between the <StructuredBody> element and the <ClinicalDocument> elements. The elements in the body can be coded.

The elements primarily seen in the header of the document are as follows:

- **<id>** This is unique for every document. It identifies each instance of the clinical document.
- **<code>** The value in this element tag specifies the kind of document , such as whether it is a discharge summary. The codes are obtained form the Logical Observation Identifiers, Names and Codes (LOINC).
- **<title>** This indicates the title of the CDA document.
- **<effectiveTime>** The purpose of this tag is to indicate the time of creation of the document
- **<confidentialityCode>** This defines the authorization constraints on the document. The value is for the entire document, unless it is overridden by a nested value. The possible values this tag can take are N, R and V. The former is the Normal code. It allows all authorized individuals to access the document. R stands for Restricted access and it allows only Health care providers, currently related to the patient, to access the document. V stands for Very restricted and allows only individuals cleared by the Privacy officer of the patient to view the document.
- **<languageCode>** This tag specifies the language of the characters in the document.
- **<setId>** Indicates if the document is a revision of an existing document. The value remains the same for all versions of the same document.

- **<versionNumber>** This indicates if the document is a revised version of a previously existing document. Each time the document is revised, the value in this element is incremented.
- **<authenticator>** Indicates if the document has been authenticated. It consists of child tags such as <signatureCode>, which indicates if the document has been signed.
- **<legalAuthenticator>** This indicates, if the document has been legally authenticated.
- **<informationRecipient>** Indicates the individuals or organizations who should receive the document. Further, it classifies the recipients as primary or secondary.
- **<author>** Indicates the creator of the document. This can be human or a machine.
- **<custodian>** This indicates the organization that creates and is responsible for the document . There can be only one custodian for the CDA document.
- **<dataEnterer>** This element tag gives information about the transcriptionist, which is the person, who converted the dictated report into text format.
- **<responsibleParty>** This represents the party who is primarily responsible for the document. It can be a person or an organization. The person is legally responsible, more so than the <legalAuthenticator>, who just authenticates the document.
- **<recordTarget>** It indicates the medical record that is the scope of the CDA document.
- **<informant>** This element tag provides information about the person who provides information about the patient. For instance this could be a family member of a comatose patient
- **<encounter>** This provides information about the encounter such as location.

The body of the CDA document can either be structured or can be comprised of an unstructured blob of data. The body of a CDA document comprises of sections. The primary element tags found in them are as follows

- **<id>** The value in this tag is used to uniquely identify the section within the CDA document.
- **<code>** This indicates the type of section, for example it indicates if it is an assessment section. The codes are obtained from the Logical Observation Identifiers, Names and Codes (LOINC)

- **<confidentialityCode>** This tag indicates the security restrictions for the section. The restrictions can allow ‘normal viewing’ or ‘restricted’ viewing or ‘very restricted’ viewing of the document, similar to the confidentialityCode tag in the header of the document. However this overrides the values in the header of the document for that particular section.
- **<text>** This includes the narrative text or actual medical report of the patient.
- **<languageCode>** This indicates the language of the character data.
- **<content>** This contains a string of text that can be uniquely referenced. It also allows recursive nesting of text values.
- **<link>** This allows referencing of multimedia content that is not part of the CDA document and works similar to a HTML anchor tag.
- **<delete>** This indicates any text that need to be deleted form the previous version
- **<insert>** This indicates text that was not present in the previous version and needs to be inserted into the present document.
- **<paragraph>** This allows the narrative content of the CDA document to be expresses as a logically structured document. This can include an optional caption.
- **<list>** This can be used to list text items
- **<table>** It allows presentation of data in a table format.
- **<caption>** This works like a header label for the paragraph, list and table element tags.
- **<subject>** This indicates the primary focus for each section of the CDA document. For example, the patient is the subject of the CDA document.
- **<component>** This allows nesting within sections. This is necessary if the context of one section is similar to another section.

```
-->
<id extension="c266" root="2.16.840.1.113883.3.933" />
<code code="11488-4" codeSystem="2.16.840.1.113883.6.1" displayName="Consultation note" />
<title>Good Health Clinic Consultation Note</title>
<effectiveTime value="20000407" />
<confidentialityCode code="N" codeSystem="2.16.840.1.113883.5.25" />
<setId extension="BB35" root="2.16.840.1.113883.3.933" />
<versionNumber value="2" />
- <legalAuthenticator>
  <time value="20000408" />
  <signatureCode code="S" />
  - <assignedEntity>
    <id extension="KP00017" root="2.16.840.1.113883.3.933" />
    - <assignedPerson>
      - <name>
          <given>Robert</given>
          <family>Dolin</family>
          <suffix>MD</suffix>
        </name>
      </assignedPerson>
    - <representedOrganization>
        <id extension="M345" root="2.16.840.1.113883.3.933" />
      </representedOrganization>
    </assignedEntity>
  </legalAuthenticator>
- <author>
  <time value="20000407" />
  - <assignedAuthor>
    <id extension="KP00017" root="2.16.840.1.113883.3.933" />
    - <assignedPerson>
      - <name>
          <given>Robert</given>
          <family>Dolin</family>
          <suffix>MD</suffix>
        </name>
      </assignedPerson>
```

Sample CDA Release Two document Header

```
      <code code="10164-2" codeSystem="2.16.840.1.113883.6.1" codeSystemName="LOINC" />
      <title>History of Present Illness</title>
    - <text>
        Henry Levin, the 7th is a 67 year old male referred for further asthma management. Onset of asthma in his
        <content revised="delete">twenties</content>
        <content revised="insert">teens</content>
        . He was hospitalized twice last year, and already twice this year. He has not been able to be weaned off
        steroids for the past several months.
      </text>
    </section>
  </component>
- <!--

    ******************************************************
      Past Medical History section
    ******************************************************
  -->
- <component>
  - <section>
      <code code="10153-2" codeSystem="2.16.840.1.113883.6.1" codeSystemName="LOINC" />
      <title>Past Medical History</title>
    - <text>
      - <list>
        - <item>
            <content ID="a1">Asthma</content>
          </item>
        - <item>
            <content ID="a2">Hypertension (see HTN.cda for details)</content>
          </item>
        - <item>
          - <content ID="a3">
              Osteoarthritis,
              <content ID="a4">right knee</content>
            </content>
          </item>
        </list>
```

Sample CDA Release Two document Body

3.1.3 Differences between CDA Release One and CDA Release Two

CDA Release Two is more RIM (Reference Information Model) derived and more structured than the previous release. It also allows encoding of values. The differences in the XML element tags listed in the HL7 Clinical Document Architecture Release Two Committee Ballot #02, Dec 08, 2003 are as follows

CDA, Release One XML Element Name	CDA, Release Two XML Element / Attribute Name
Levelone	ClinicalDocument
clinical_document_header	--doesn't exist--
Id	Id
set_id	SetId
version_nbr	VersionNumber
document_type_cd	Code
service_tmr	EffectiveTime
origination_dttm	--doesn't exist--
Copy_dttm	CopyTime
confidentiality_cd	ConfidentialityCode
document_relationship	RelatedDocument
document_relationship.type_cd	TypeCode
Related_document	ParentDocuments
Fulfills_order	RelatedOrder
Fulfills_order.type_cd	TypeCode
Order	Order
patient_encounter	Encounter
practice_setting_cd	Code
encounter_tmr	EffectiveTime
Service_location	HealthCareFacility, Place
Addr	Addr
Authenticator	Authenticator
authenticator.type_cd	TypeCode
participation_tmr	Time
signature_cd	SignatureCode
Person	AssignedPerson
Person_name	Name
effective_tmr	ValidTime
Nm	--not present--
Person_name.type_cd	"use" attribute on <name>
Telecom	Telecom
Legal_authenticator	LegalAuthenticator
legal_authenticator.type_cd	TypeCode
Intended_recipient	InformationRecipient

CDA, Release One XML Element Name	CDA, Release Two XML Element / Attribute Name
intended_recipient.type_cd	TypeCode
Originator	Author
originator.type_cd	TypeCode
originating_organization	Custodian
originating_organization. type_cd	TypeCode
Organization	RepresentedOrganization
organization.nm	Name
Transcriptionist	DataEnterer
transcriptionist.type_cd	TypeCode
Provider	responsibleParty, encounterPerformer
provider.type_cd	TypeCode
function_cd	FunctionCode
Service_actor.type_cd	TypeCode
Patient	recordTarget, subject
Patient.type_cd	TypeCode
is_known_by	Id
Is_known_to	ProviderOrganization
birth_dttm	BirthTime
administrative_gender_cd	AdministrativeGenderCode
originating_device	Author
originating_device.type_cd	TypeCode
Device	Device
Responsibility	Maintainer
responsibility.type_cd	ClassCode
responsibility_tmr	EffectiveTime
Service_target	Participant
Service_target.type_cd	TypeCode
Body	StructuredBody
Section	Section
non_xml	NonXMLBody
Content	Content
Link	Link
link_html	LinkHtml
Coded_entry	CodedEntry
Coded_entry.id	Id
Coded_entry.value	Value
observation_media	ObservationMedia
observation_media.id	Id
observation_media.value	Value
local_markup	--doesn't exist--
local_header	--doesn't exist--
local_attr	--doesn't exist--

CDA, Release One XML Element Name	CDA, Release Two XML Element / Attribute Name
Paragraph	Paragraph
List	List
Item	Item
Table	Table
Caption	Caption
caption_cd	LocalCaptionCode
other table elements	--unchanged--

Source: HL7 Clinical Document Architecture Release 2.0 – Liora Alschuler,Robert H.Dolin,Sandy Boyer, Calvin Beebe, Paul V. Biron,Fred Behlen

3.2 XML Databases

XML is the generally recognized text format used by all applications for storage and exchange of data that is language and platform independent. The widely accepted usage of XML has prompted a number of commercially available Database Management Systems to support storage, retrieval and manipulation of XML data. XML can be stored in databases in basically two forms. One is by mapping the XML elements and attributes to the underlying storage schema of the database. The other means is by storing the entire XML document as a whole unit in the database. This is also referred to as native XML databases.

3.2.1 XML Enabled Databases

Mapping of an XML document to a database schema can follow either the relational mapping or object relational mapping methods. However, the mapping does not provide for storage of comment statements, processing instructions and encoding statements. Further, mapping the document to a relational database schema involves mapping the element names or attributes to table columns. If the element tags consist of a number of child tags, then each such element could be considered as a table and the child tags could be mapped as columns for that table. Some of the more commercially popular database management systems allow the mapping to be done via an XML Schema definition (XSD). An XSD allows the definition of mapping of element names or attribute names to tables in the relational database. Thus the XSD can be applied to every XML document of the same format to intrinsically map all the tags to the underlying relational database schema.

Object-relational mapping allows mapping of semi-structured XML documents. The XML document is constructed as a tree of objects. All complex element tags that contain attributes child elements or values are modeled as classes. Element tags, which contain only PCDATA, can be modeled as scalar objects. Object views can then be used to map the object relational databases to relational tables, such that all classes are mapped to tables and all scalar objects are mapped to columns in the tables.

3.2.2 Native XML databases

Native XML databases allow storage of the entire XML document without modification in the database. The fundamental unit of data storage for such databases is XML. These databases allow storage and retrieval of structured, semi-structured and unstructured XML documents in the database. Native XML databases can also show a marked improvement in performance, compared to relational XML databases. This is because, unlike relational databases, native XML databases do not need to perform any joins between tables to retrieve query results. All the data is stored together physically as one unit. Further, since native XML databases allow storage of the entire document as XML, it facilitates the usage of XML queries to manipulate the data.

Native XML databases can be considered to be of two types. One is Text based and the other is Model based. Text based native XML databases store the XML document as text. If this were to be implemented in a relational database, the document would be stored as a Character Large Object (CLOB) of data. The advantage of text-based databases is that, they contain indices to the document. This tremendously improves performance during query of the document. Thus it can out perform a relational database for retrieval of query results.

Model-based native XML databases do not save the document as text. Instead, an object model is created for the document and that model is stored in the database. The model can be stored in a relational or object oriented database. The performance will depend on the underlying storage mechanism used for the model.

The features available in Native XML databases can be summarized a follows:

- Native XML databases stored XML documents in collections. The collections are analogous to tables, and the documents can be considered to be rows within those tables. A hierarchy of collections can be created in addition to nested collections.

- Most of the commercially popular Native XML databases support the use of XPath for data manipulation (query) in the XML documents. XQuery is also supported in some of the native XML databases.
- Insertion and deletion of XML documents is made very simple. The entire document can be deleted, like a row in a table.
- Indexing provided for the documents provide improved query performance.

3.3 XQuery

XQuery is a language that can be used to query XML documents. It is analogous to Structured Query Language (SQL), which is used to query databases. XML is the standard followed by most developers today for information interchange. The increase in the use of XML format has led to the need for a means to query and update these documents. Using XQuery, one can make searches on XML documents and can extract values of element nodes or attribute values of nodes. It is particularly useful for Native XML databases, where the data is stored in the database in the form of XML.

3.3.1 Applications

XQuery can be used for performing functions, which are far more advanced than the ordinary search and update capabilities. It provides scope to transform the content and structure of an XML document using element and attribute constructors. Data integration can also be performed, by combining the results of different queries. Thus XQuery, in addition to providing results of queries, can also provide a means for presentation of the data. Some applications are listed below:

- XQuery is not limited to XML documents only. It can be applied to text documents too. Application log files, for instance, contain large amount of potentially useful information in text format. XQuery facilitates mining of information from these files by mapping the data to an XML data model.
- XQuery allows easy integration of results from different database sources, so that the results can be presented in one format. This is particularly useful for reporting applications, which need to work with multiple and varied data sources.

- XQuery allows transformation of query results. This is useful for applications, which need the result set to be in a format suitable for input.
- XQuery can be used to search for documents. For example, trade documents for several industries, need to be saved in their original form. Finding the required document later might prove cumbersome. This can be resolved using XQuery
- XQuery provides dynamic indexes for documents. It also allows querying embedded data in documents, such as clinical documents.

3.3.2 Requirements

W3C Working group has stipulated that the following requirements be met for XQuery:

- The syntax requirements need XQuery to be human readable and allows it to have more than one syntax binding
- XQuery needs to be declarative and not evaluative
- XQuery needs to be independent of any protocols, with which it might be used
- All error conditions need to be standardised. Some possible errors are processing errors within expressions or those generated by external functions.
- Ability to update capabilities in future versions must be included.
- XQuery should be defined for finite instances of the data model.
- XQuery should support operations on all data types in the XML Query Data Model.
- XQuery should function, even in the absence of schemas.
- Any operation on collections should include universal and existential quantifiers.
- XQuery should be capable of combining results from multiple sources.
- XQuery should be able to preserve hierarchy of input structures and should be able to perform sorting on the results.
- XQuery should be able to handle external and internal document references.

3.3.3 XQuery Expressions

An XQuery expression can be used to retrieve data values from an XML document. The XQuery expressions explained in this section will retrieve values from nodes in the XML document called “subject.xml” listed below.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <sub>
        <title>math category=”calculus”</title>
        <tutor>
              <last>Randy</last>
              <first>W.</first>
        </tutor>
  </sub>
  <sub>
       <title>math category=”algebra”</title>
       <tutor>
             <last>Reynolds</last>
             <first>Mandy</first>
       </tutor>
  </sub>
  <sub>
        <title>math category=”calculus”</title>
        <tutor>
             <last>Steven</last>
             <first>Richard</first>
        </tutor>
  </sub>
  <sub>
           <title>science category=”biology”</title>
           <tutor>
                 <last>Rex</last>
                 <first>Walden</first>
           </tutor>
  </sub>
```

An XQuery expression consists of XML fragments, such as elements and attributes and returns results in the form of XML. It can consist of XPath expressions. An XPath expression can consist of either the path of the node from the root or the path of the node in that context and would retrieve values for the node in that expression. For example consider the following expression:

doc (“subject.xml”)/sub/title[category=”calculus”]

In the above expression, the function ‘doc’ opens the document node of the XML document. The path sub/title [category=”calculus”] extracts all ‘title’ nodes, which have ‘calculus’ as the attribute value for the attribute ‘category’.

An XQuery expression can also consist of a FLWOR expression, which stands for For, Let, Where, Order by, Return. A FLWOR expression allows the construction of complex queries and is analogous to the Select statement in SQL. Consider the following FLWOR expression

for $x in doc("subject.xml")/sub/title where $x//title=”math”order by $x//last return $x//title

In the above expression, ‘for’ places all ‘title’ elements in the variable ‘x, while ‘where’ selects only those nodes where the title value is ‘math’. ‘order by’ orders the nodes , according to the value in ‘last’ and the values in ‘title’ are returned. Here the path /sub/title is the same as //title. More than one ‘for’ clause can be used, to create a nested XQuery.

XQuery can also consist of functions such as count() , name() etc. The former returns a count of the number of nodes in a node set, while the latter retrieves the name of the node. An XQuery expression can also consist of operators such as relational and logical operators.

4. SYSTEM OVERVIEW

This chapter gives a brief overview of the Data Mining Framework. The System Architecture, goals, benefits and evolution are described in detail.

4.1 Goals

The main objectives of the Data Mining Framework are summarized below:

- *Data Abstraction*

The designed system should allow 'abstraction' or extraction of information from the Clinical Document Architecture (CDA) files. A means to store the abstracted data for easy retrieval and manipulation at any desired time should also be devised.

- *Query Data*

The Data Mining Framework should allow queries to be run against the data in the CDA documents. Facility to query against a collection or sub collection of documents should also be provided. The queries should be structured, such that meaningful information can be retrieved from the documents to improve the quality of health care.

- *Anonymous Data*

The CDA documents contain vast amounts of patient information, which can be used to improve the quality of health care. However, in addition to carrying information about the patient diseases, these documents also contain details about the patient, affecting his privacy. A facility to eliminate all such identifying information about the patient needs to be provided.

- *Flexibility*

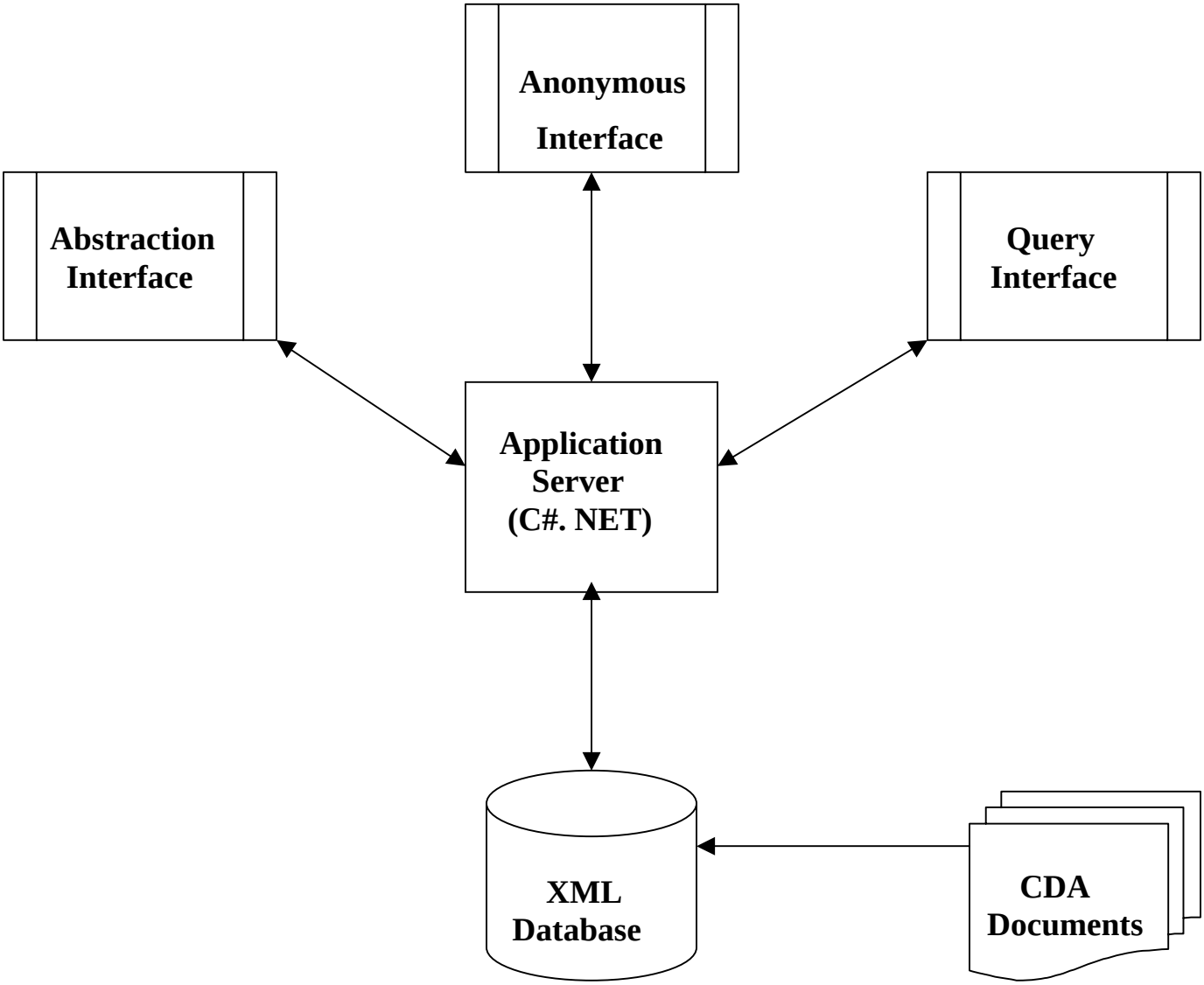
The Data Mining Framework should be flexible to work with all versions of CDA documents. At the time of this documentation, CDA Release One and CDA Release Two were the possible means to represent clinical information. The Data Mining Framework needs to functions with both these Releases. Further, flexibility as regards to the information that needs to be abstracted should be provided.

- *Scalability*

The Data Mining Framework should be created as individual components. This would allow further modifications to be easily incorporated into the application and thereby make it scalable.

4.2 System Architecture

The following diagram depicts the architecture of the software component of the Data Mining Framework



Each of the components in the system is explained below:

- *Abstraction Interface*

This is a web interface and is concerned with extracting information from the clinical documents and presenting the results to the user. The user identifies the documents from which the data need to be abstracted using this interface. This information is passed on to the application server module. The output generated from the application server is presented to the user in the Abstraction interface

- *Query Interface*

The Query interface is a graphical user web interface, concerned with querying the database. The user can run a query by choosing from the options provided on this web interface. The choice is passed on to the application server module, which interacts with the database and retrieves results. These results can then be displayed in the query interface.

- *Anonymous Interface*

This web interface is concerned with making clinical data anonymous. It allows the user to choose the clinical documents, which need to be made anonymous and represents the clinical information in textual format, allowing the user to make appropriate changes. The user can save the changes to the database using the same interface. The application server module handles the interaction with the database.

- *Application Server*

The application server module is the main functional module of the system. It consists of code to manipulate the values in the CDA documents, irrespective of whether it is CDA Release One or Release Two documents. It handles the representation of XML values in web form components such as textboxes and the storage of values as XML documents in the database. In short, this module is responsible for the interaction between the web interfaces and the database management system.

- *XML Database*

This database management system is responsible for storage, retrieval and manipulation of the CDA documents. Since the CDA documents are encoded in XML, the chosen database is an XML database. A native XML database, i.e. a database where XML is the fundamental unit of data storage is preferable, since all the clinical data is in XML format.

- *CDA documents*

The CDA documents encoded in XML contain the clinical data, which needs to be mined. The Data Mining Framework is designed to work with CDA Release One and Release Two documents. These documents are stored in the XML database.

4.3 System Evolution

The software life cycle model employed for designing the Data Mining framework was the Object-Oriented life cycle model. This model allows for iteration between different phases of development such as analysis, design and implementation phases. In addition it drastically reduces the maintenance phase, due to the object-oriented paradigm of overlap between the phases.

Achieving flexibility in the system design is one of the more significant goals of the Data Mining Framework. Flexibility is to be achieved with regard to the versions of CDA documents used, the type of data abstracted and the kind of queries run against the database. Hence the requirements phase for this application would overlap with the other phases of the software development cycle. Thus the object-oriented life cycle is best suited for the Data Mining Framework.

In addition to supporting parallelism of phases, the object-oriented model supports incremental development of the application. This allows different modules of the application to be integrated at different phases of time, to provide system functionality. Consequently, in the Data Mining Framework, the different modules like abstraction and query modules could be integrated and implemented in different phases.

4.4 Constraints

The chosen environment for the application development of the Data Mining Framework is Visual Studio. NET (VS. NET). At the time of this documentation, VS. NET was designed to function only for the Windows operating system. Hence the platform for the Application Server must be a Windows platform. However, the clients could access the functionalities of the system using a web browser and hence the platform used for the client system is not significant. VS.NET was chosen for its high level of support for web forms and web services.

4.5 Benefits

The facilities provided by the Data Mining Framework are summarized below:

- It allows extraction of information from clinical documents and the ability to run queries against the clinical data.
- It de-identifies patient information.
- It provides access to the application from remote machines using browsers.
- It provides flexibility to work with different versions of CDA documents.
- Changes can be made to the kind of information, which is to be extracted from the clinical documents.
- The object-oriented life cycle followed, ensures that the maintenance cost would be reduced.

5.DESIGN

This chapter describes the design methodology and constraints of the Data Mining Framework. Detailed UML diagrams analyze the working of the system.

5.1 Description:

The Data Mining Framework allows the storage, retrieval and management of reports generated at each encounter of a patient. All patient encounter information is represented in Clinical Document Architecture (CDA) format. CDA documents, which is encoded in XML, is a HL7 standard for representing discharge summaries and progress notes of the patient. The revised CDA specification (CDA Release Two) announced by the HL7 committee is more structured than CDA Release One. It also differs from CDA 1.0 in the element tags. The Data Mining Framework provides flexibility to work with both CDA 1.0 and CDA 2.0 documents. In addition to the management of the information, the Data Mining Framework provides web-based user-friendly interfaces to interact with the data, which are Anonymous Data, Data Abstraction and Data Query.

Patient encounter reports can consist of sensitive information, such as names and other personal information of the patients and doctors. However, the Health Insurance Portability and Accountability Act (HIPAA) regulation requires that patient confidentiality be preserved. The Anonymous Data Interface module is concerned with protecting the privacy of the patient and Health Care Personal. It displays the XML document in a web form and allows masking of sensitive information. Changes are reflected in the database.

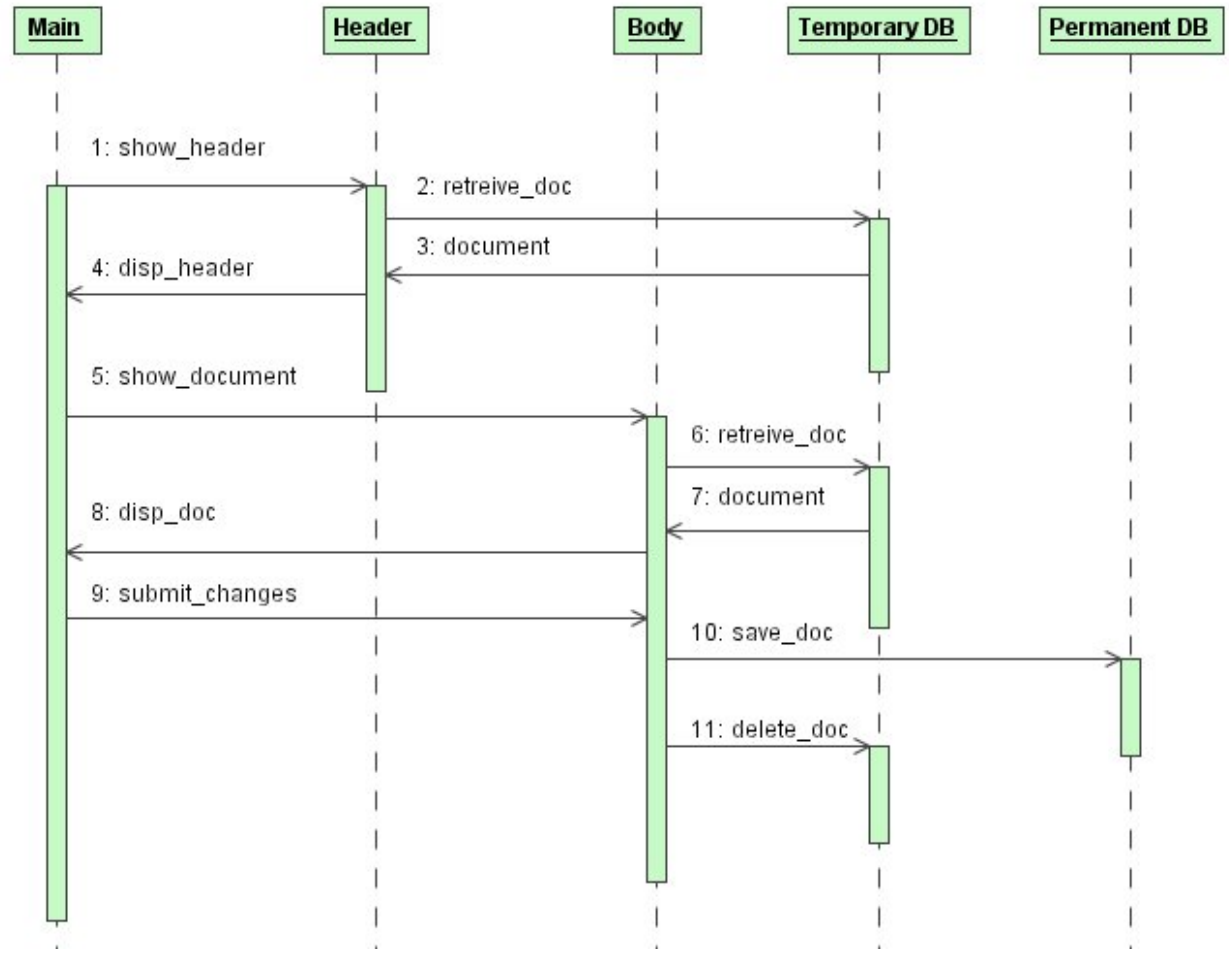
Abstraction is the analysis of a document to extract relevant information. The information is mapped on to a structured form, so that it can be loaded into the database. The American Hospital Association (AHA), The Federation of American Hospitals (FAH) and the Association of American Medical Colleges (AAMC), require 10 quality measures from medical reports to be submitted as part of a national voluntary initiative The purpose of this, is to improve patient quality care and standardize data collection mechanisms. The Data Abstraction module is concerned with the manual abstraction of these quality measures form encounter reports. A web-based interface is provided for the medical transcriptionist to identify the quality measures. The measures are then mapped to an XML form and loaded into the database.

The Data Query module provides a user-friendly web interface to query the XML database. It determines patterns to obtain useful information from the encounter reports and the extracted quality measures. For example, a query could be initiated to determine the number of Acute Myocardial Infarction (AMI) patients who had been administered aspirin on arrival at the hospital.

5.2 UML Diagrams

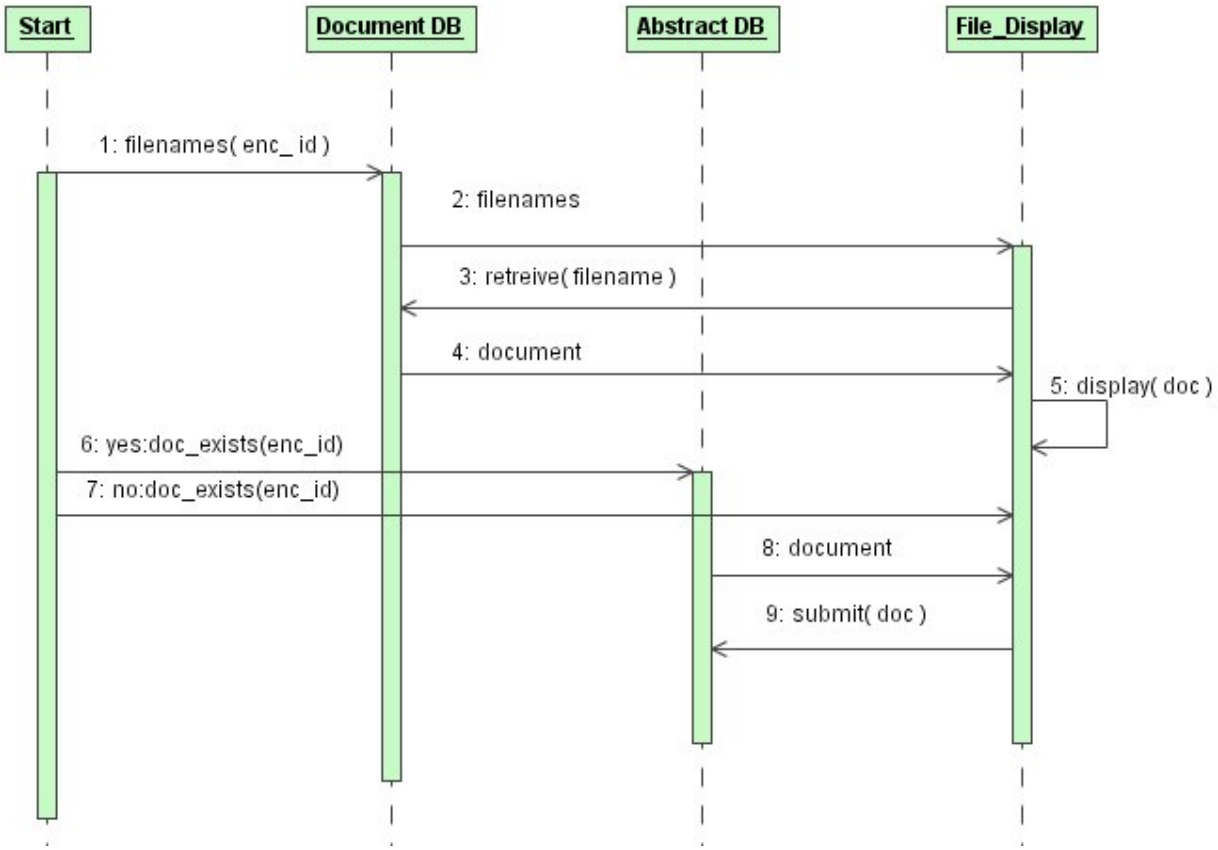
5.2.1 Sequence Diagrams

Anonymous Data Module:



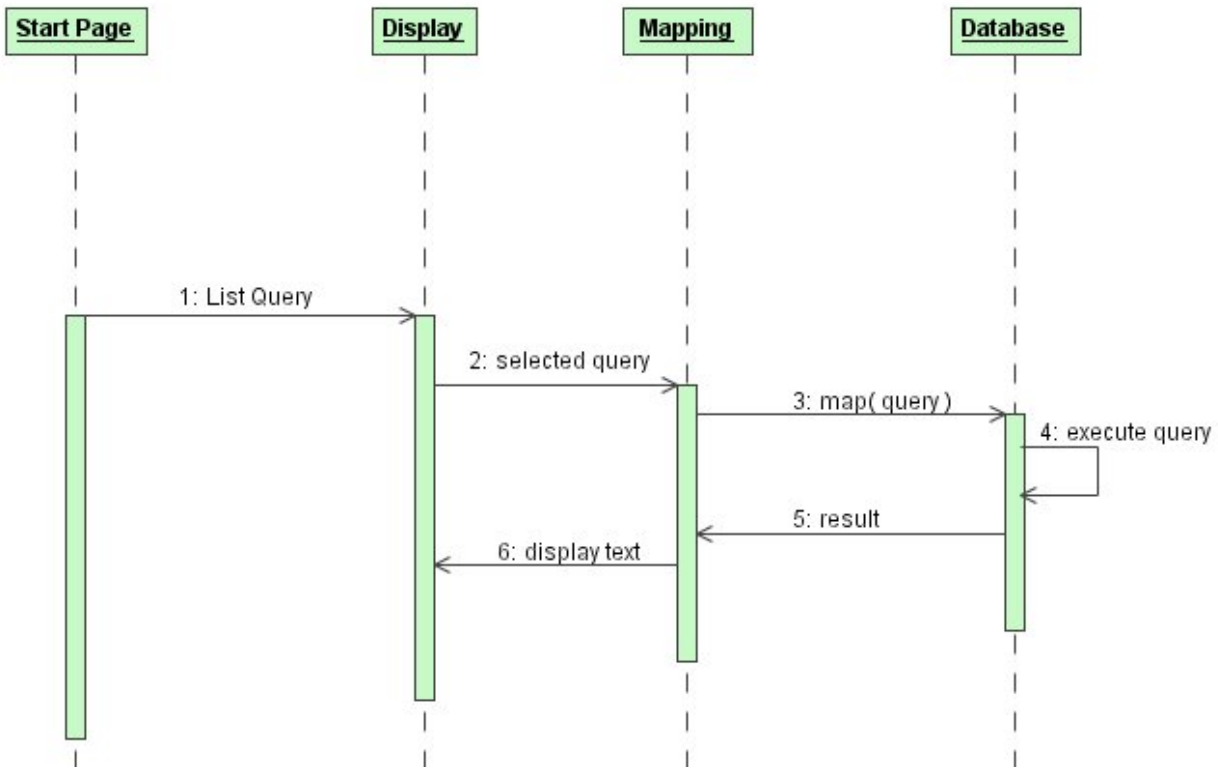
The Anonymous Data Module consists of a Main page frame in which the header and body of the documents, which need to be de-sensitized, are displayed. All these documents are stored in a temporary database and the first document in the temporary database is displayed on the Main page. The Main page calls the Header and Body Forms. The Header Form is responsible to display the Header information in the main page. It retrieves the XML document from the temporary database, and displays the header on the Web Form in a table, specifying the attribute name and value. Similarly the Body Form is responsible for displaying the content of the encounter reports in modifiable text boxes on the main page. After it has been de-sensitized, it is entered into the permanent database. The document is then deleted from the temporary database, so that next document can be retrieved for de-sensitization.

Data Abstraction Module:



The Abstraction module allows manual abstraction of the ten quality measures specified in the National Voluntary Hospital Reporting Initiative from all the reports of a given encounter of a patient. An encounter id is specified in the Start Page. All encounter reports having the specified encounter id are retrieved from the Document database and the filenames are displayed on the File Display page. Any of the filenames listed can be selected. A selection prompts a request to the database and retrieves the document from the database for display on the File Display Page. A request is also made to the Abstract database to retrieve the abstracted document for that encounter id, if it exists. In the event that a file exists, it is displayed on the File Display page in editable text fields. In the event that it does not, empty text fields to enter information are displayed. The quality measures can be manually abstracted from any of the files listed on the File Display page, by selecting the appropriate file name. The abstracted measures created are then converted to XML form and saved to the database.

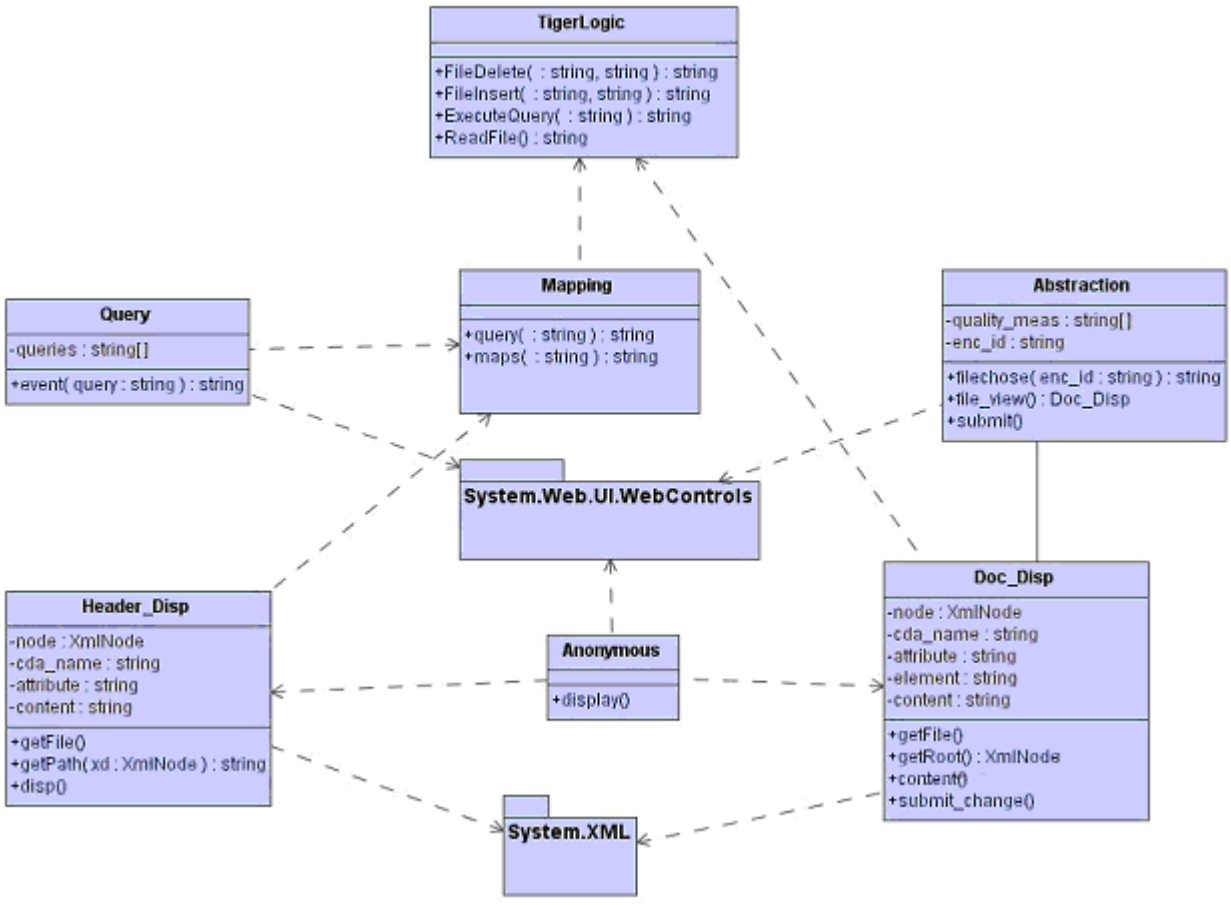
Data Query Module



The Query module allows an authorized user to query the database of encounter reports. A list of predefined queries is presented to the user in a list. The user selects the query from the given list. A query is constructed in the mapping class in XQuery. The XQuery is run against the database. The CDA files against which the XQuery is to be run, is mentioned in the XQuery itself. The results are returned in the form of text and then can be displayed in the display page.

5.2.2 Class Diagram

The following is the class diagram for the Data Mining Framework:



5.3 Classes and Functions

The class diagram clearly shows the relationship between the classes in the Data Mining Framework. The user is presented with a Web Interface and is allowed a choice among abstracting, querying and creating anonymous data, with each choice making a call to the respective class.

5.3.1 Anonymous class:

The Anonymous class displays the results of the Header_Disp and Doc_Disp classes. The Header_Disp class is responsible for displaying the header values of the CDA document. The attribute names in the header are in the form of abbreviations, which are difficult to understand, unless the structure of the CDA document is known. Hence the attribute names are mapped to new names, which can be more easily understood, before they are displayed. Thus the Header_Disp class is responsible for displaying the header attributes and their values. This class works irrespective of whether the document is CDA Release One or Release Two. The following are the functions in the Header_Disp class:

- *getFile()*: This function is responsible for getting the name of the CDA file from the database. All CDA files are placed in a temporary collection, before the records are made anonymous. This function gets the name of the file, which is top most in the temporary collection an opens an input stream to that file. Once a file had been made anonymous, it is deleted form the collection. Hence this function gets a new file name each time, if the previous call to this class, created an anonymous file.
- *disp()*: This function is responsible for displaying the attribute names and values. An XmlNode is used to read the element values. All possible nodes such as comment nodes, empty nodes, text nodes etc are taken into consideration. This method is flexible to work with both CDA Release One and Release Two.
- *getPath(XmlNode)*: This function is responsible for producing the masked name for the attribute name in the CDA Header. It finds the path of the attribute name in the XML hierarchy and this is used to map the name

The Doc_Disp class is responsible for displaying the body part of the CDA document. The element values are to be displayed in the form of text in a graphical user web interface, in the form of modifiable text boxes. The functions in this class are as follows:

- *getFile():* This is analogous to the function in the Header_Disp class and is responsible for getting the name of the topmost file in the temporary collection and opening an input stream to that document.
- *getRoot()*: This function gets the root node of the body of the CDA document. Once this node is obtained, display of text values can proceed from this node onwards down the hierarchy.
- *content()*: This node is responsible for displaying all the values in the element nodes. Attribute values, in addition to text values need to be displayed. Nodes such as comment nodes and empty nodes have to be skipped. Moreover, it should work for CDA Release One and CDA Release Two documents.
- *Submit_change()*: Once the CDA document has been made anonymous by the user in the graphical user interface, the changes are saved using this function. The CDA document is reconstructed from the values in the textboxes and the result is saved along with header as a CDA document in a new collection.

5.3.2 Abstraction class

This class is responsible for abstracting the quality measures, known as the Hospital Quality Alliance measures. The quality measures are abstracted from the entire encounter reports generated for the patient and saved as an XML document. The encounter reports are retrieved based on the encounter ID of the patient. A manual abstraction of the reports is prepared by an authorized personal and the resultant document is stored in the database in the form of an XML document. The following are the functions in this class:

- *filechose(string)*: This function retrieves all the encounter reports associated with patient. It takes the encounter ID of the patient as input. A query is run against the database to retrieve all the CDA documents related to that encounter and the filenames are listed to the user in a graphical user web interface.
- *file_view ()*: This function allows the user to view any of the listed files he chooses. This function uses the ‘content’ function of the Doc_Disp class to display the body of the

document in text boxes. The quality measures can be manually abstracted by choosing to view each of the files in turn.

- *submit()* : This function saves the abstracted quality measures. A new XML document is created and is saved in the database.

5.3.3 Query class

This class allows the user to query the CDA documents. A pre-defined set of queries is created for the user to choose from. The selected query is run against the database of CDA documents and the results are displayed in the web interface. The following functions are present in this class:

- *event(string):* This function retrieves the choice of the user. The 'query' function of the Mapping class is called and a query is run against the database. The results that are retrieved are displayed to the user in the form of a graphical web interface.

5.3.4 Mapping class

This class consists of a set of functions that are used by the other classes. The following are the functions in this class:

- *query(string)* : This function is called by the 'Query' class. Depending on the user choice, a particular query is run against the database. The retrieved results are returned in the form of a string to the calling function.
- *maps(string):* This function is used by the Header_Disp class. It allows the mapping between the attribute name in the CDA header with a new name. The path of the attribute of the XML Node is obtained using the 'getPath' function in that class. This path is used by the 'maps' function to map the attribute name to a new name.

5.3.5 TigerLogic class

This class contains the class made to the database, which is the TigerLogic XDMS. The following functions are found in this class:

- *FileDelete(string,string):* This function allows the deletion of a CDA document from the database. It accepts two input parameters. One is the file name and the other is the name of the collection, which contains the file.

- *FileInsert(string,string):* This function allows the insertion of an XML document into the database. It accepts two input parameters, viz. the name of the file to be saved and the collection, into which it is to be placed.
- *ExecuteQuery(string):* This function executes an XQuery against the database. The input parameter accepted by this function is the XQuery.
- *ReadFile():* This function allows the user to open an input stream to the CDA File. This allows the CDA document to be read, so that it can be displayed by methods in other classes, in a graphical web interface.

6. IMPLEMENTATION

This chapter describes the programming tools that were used to implement the Data Mining Framework. It includes a comparison of the available tools, to highlight the benefits of the chosen tool.

6.1 Database Management System:

The Database Management System should have high-level support for storage and manipulation of XML documents. The Data Mining Framework requires a robust Database Management System that implicitly supports retrieval and insertion of different versions of Clinical Document Architecture documents. Ability to query the element tags is also required. Based on their support for XML documents, SQL Server, Oracle and TigerLogic Database Management Systems were considered.

6.1.1 SQL Server 2000

SQL Server allows storage of XML data in relational tables using annotated XML Schema definitions (XSD) .The annotated XSD describes the contents and format of the XML documents and maps the XML element tags to the relational tables. An abstraction layer, called an XML view exposes the relational tables as structured, hierarchical data. The data can also be addresses using SQL query syntax using OPENXML keyword. This provides an updateable rowset for the XML documents

Retrieval of data in the form of XML is possible by using the FOR XML key word. SQL Server allows the use of AUTO, RAW and EXPLICIT extensions to this key word. RAW is the simplest form of XML retrieval. In this each row of the rowset is represented as an XML element, using the identifier ‘row’. All non-NULL column values for that row are represented as attributes for that row. The AUTO mode allows retrieval of query results in a nested form. Each table in the SQL query is represented as an XML element, with the columns listed as attributes or sub elements. The EXPLICIT mode is the most complex of the three modes. It allows the user to explicitly state the form of the XML tree.

SQL Server uses SQL query and XPath to search for results within the XML document. It does not support XQuery

6.1.2. Oracle 8i

Oracle allows XML documents to be stored either as a single object in a Character Large Object (CLOB), Binary Large Object (BLOB) or as data distributed over object relational tables. The former method of storing documents is useful, when the data is static and not likely to be updated. The latter method is preferred when the data is structures and likely to be queried and updated often. An alternate method of storing XML documents in Oracle is to use a combination of the methods described above. For example in a given document, the structured data could be stored in relational tables, while the unstructured data, such as comments could be stored in a CLOB or a BLOB. An XML view is used to combine the data in both the forms and present it to the user.

Oracle uses XML SQL utility to retrieve results of queries as XML. The XML tree created depends on the internal representation of the data in the database. Column names are mapped to top-level elements. Objects are mapped to elements containing sub-elements.

Oracle supports XPath, but does not support Xquery.

6.1.3 TigerLogic 2.0

TigerLogic is a native XML database for which the logical unit of storage is an XML node. In other words, it allows storage and retrieval of data in the form of XML nodes .No mapping between element tags and relational tables exist. Documents are stored in collections (analogous to tables in a relational database), thereby allowing query operations to be performed on nodes in a set of XML documents. Further, no DOM objects or tree structures are used to store the XML documents. Structural metadata is used for the data storage, thereby removing the need for tree traversal. Using nodes as the unit of storage ensures that the granularity of data access is fine grained. In addition, it allows the data to be accessed from the nodes, without having to load the entire document into the memory. Retrieval of data in TigerLogic is also in the form of XML nodes.

TigerLogic provides improved XML Query performance by reducing the target search space. Eliminating semi-structured and structurally irrelevant data before the query is processed can reduce the target search space. Other means to reduce target search space include enabling indexes at collection levels. The use of nodes as storage units eliminates the need to load the entire document into memory, thereby reducing query-processing time.

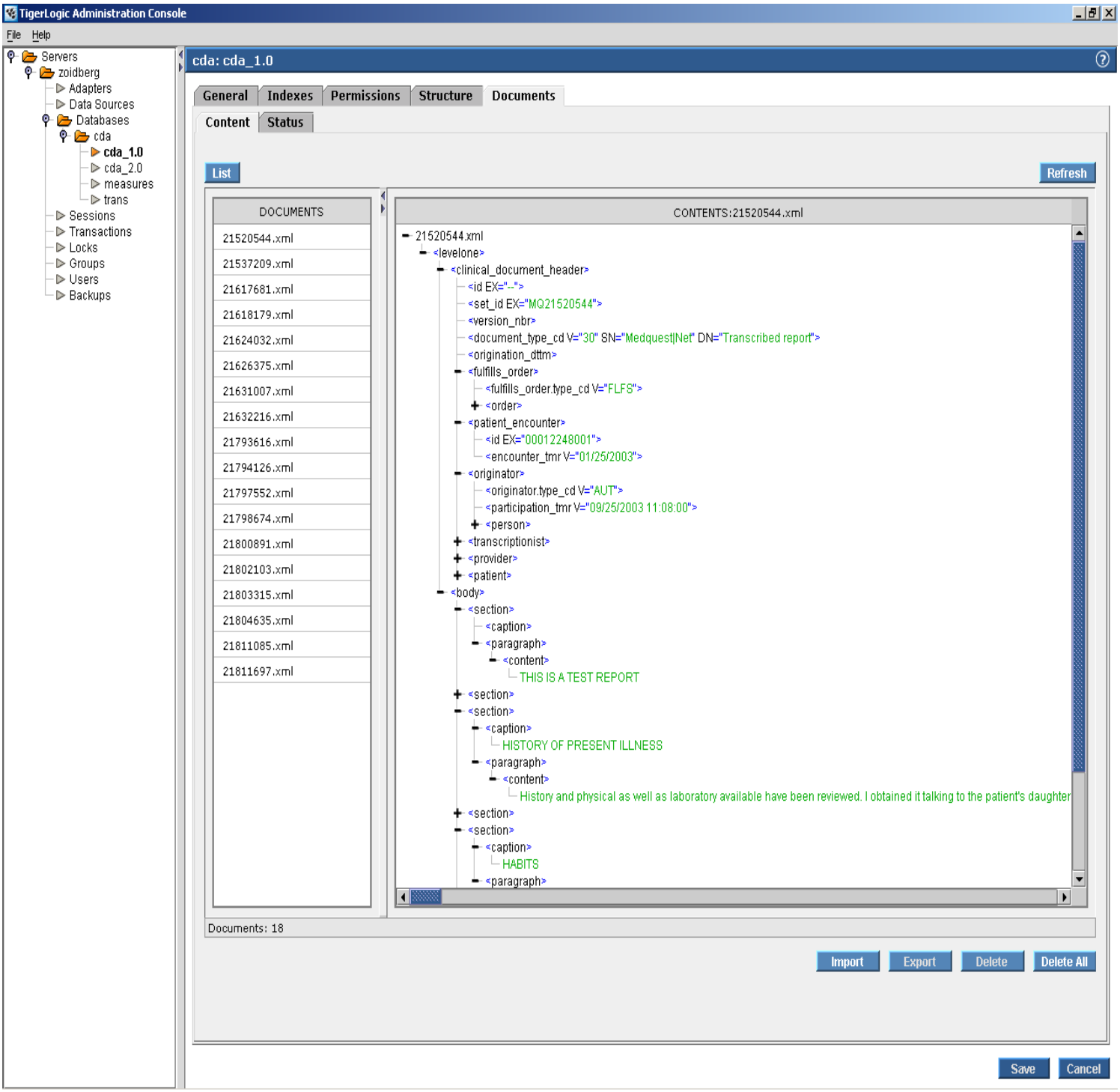


Fig 5.1: View of a Clinical Document Architecture 1.0 document in TigerLogic

6.1.4 Chosen Database Management System

TigerLogic XDMS was chosen as the Database Management System for the Data Mining Framework over SQL Server and Oracle DBMS'. The reasons are stated below:

- Data Storage representation:

  TigerLogic XDMS allows storage of data as XML nodes. This eliminates the need to use an XML Schema Definition to map the element tags to the relational tables as in the case of the other DBMS'. Using XML nodes as the storage unit also improves query performance, which is a requirement for the Data Mining Framework

- Data Manipulation language (DML) Support:

  TigerLogic supports XQuery and XPath for querying the XML documents, unlike the other DBMS', which support only XPath. TigerLogic supports the notion of storing XML Documents in a collection, so that a query can run against a set of XML Documents at a time. In addition it also supports querying across collections. The XQuery feature is useful to run complex queries in the Data Mining Framework.

- Data Retrieval:

  TigerLogic supports the retrieval of the entire document or results of queries as XML. The query results are returned as XML nodes, unlike SQL Server, where the query result structure has to be explicitly stated using the available modes in the FOR XML keyword.

- Query Performance:

  TigerLogic provides improved query performance by reducing the target search space. This is achieved by eliminating semi-structured data during storage, by enabling indices at collection level and by avoiding tree traversal and loading of entire document into the memory.

6.2 Application Development Environment:

The Development Environment used for creating the Data Mining Framework was Visual Studio.NET (VS.NET). VS.NET provides a comprehensive set of tools for developing ASP Web applications, XML Web services and mobile applications. Additionally it provides a very user friendly and powerful Integrated Development Environment (IDE). The IDE in VS.NET provides a choice of programming languages such as Visual Basic. NET, Visual C#. NET, Visual J#. NET and Visual C++. NET. All the languages support object-oriented development and use the same IDE for development. This is particularly useful, as it allows the sharing of

tools in applications, which use multiple languages, allowing cross language inheritance. Additionally they can make use of technologies provided by VS.NET, which vastly simplify the process of creating XML web services and web applications.

VS.NET provides tools for developing applications and web applications for devices such as Pockets PC's, personal digital assistants and mobile devices. Further, it provides extensive support for XML, using the XML Designer to edit and create XML Schemas. It also allows the inclusion of XML Web services. Thus VS.NET not only provides an extensive library for developing applications, but also provides the environment in which the application can be executed.

Some of the advantages of VS.NET can be summarized as follows:

- All the programming languages VS.NET supports are based on object oriented principles.
- Consists of well-designed and comprehensive libraries designed from the bottom up.
- All the languages supported by VS.NET share he same IDE and compile to a common Intermediate language. This allows interoperability and cross language inheritance.
- ASP.NET supports dynamic web pages, as it allows the code to be written in languages such as C#. NET and VB.NET
- ADO.NET allows access to data sources and databases efficiently. Access to files and directories is made simple by various additional components.
- Sharing of code is possible using an assembly. These assemblies can have different versions and can still function side by side.
- It has security features, which can restrict the use of methods and classes to only a certain group of users or processes. This also applies to assemblies.
- Developing web services is as simple as developing any other application.

This VS.NET has a high level of support for web based applications and makes the development of web services very simple. Further it has very thorough and comprehensive packages to support manipulation of XML documents, such as CDA documents. Based upon the above-mentioned advantages, VS.NET was chosen as the development environment for the Data Mining Framework.

6.3 Programming Language

The Programming language used for the development of this project was C#.NET. Among the several languages considered, C#.NET seemed the best suited for the requirements. Some of the advantages of C#.NET over the other languages is summarized as follows:

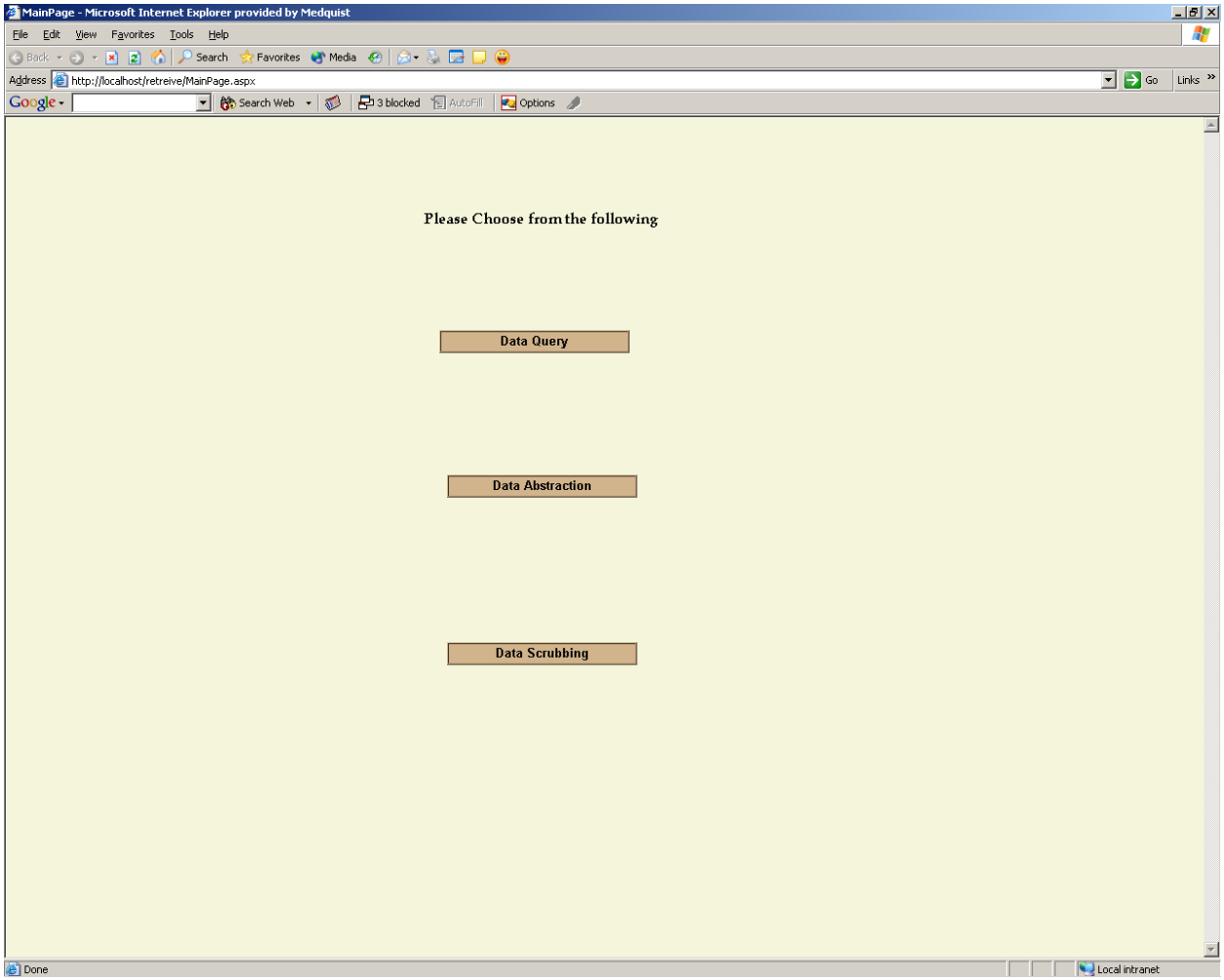
- C#.NET is an object oriented language, supporting features, such as virtual functions, inheritance and polymorphism.
- It has comprehensive libraries that allow manipulation of XML documents, such as System.XML and System.XPath
- It is as intuitive to use as Visual Basic and at the same time, has the same high performance levels and low memory access of C++.
- It contains all the .NET base class libraries, in addition to access to the Windows API.
- It supports event handling and properties, similar to Visual Basic.
- It can create dynamic ASP.NET pages.
- It is designed to work with VS.NET, unlike Java. Hence it can make use of some of the advantages of VS.NET, such as cross language inheritance.
- C#.NET is complied to an Intermediate Language. This gives it platform independence. Further, Intermediate language is Just-In-Time compiled. This leads to an improvement in the performance ratio.
- Reading and Writing Streamed XML is very easy using the available classed such as XmlReader and XmlWriter.
- It consists of several libraries, which allow easy I/O operations on files.

Thus C#.NET has several advantages over existing programming languages. Hence it was chosen as the programming language for developing the ‘Data Ming Framework’.

6.4 Web Interfaces

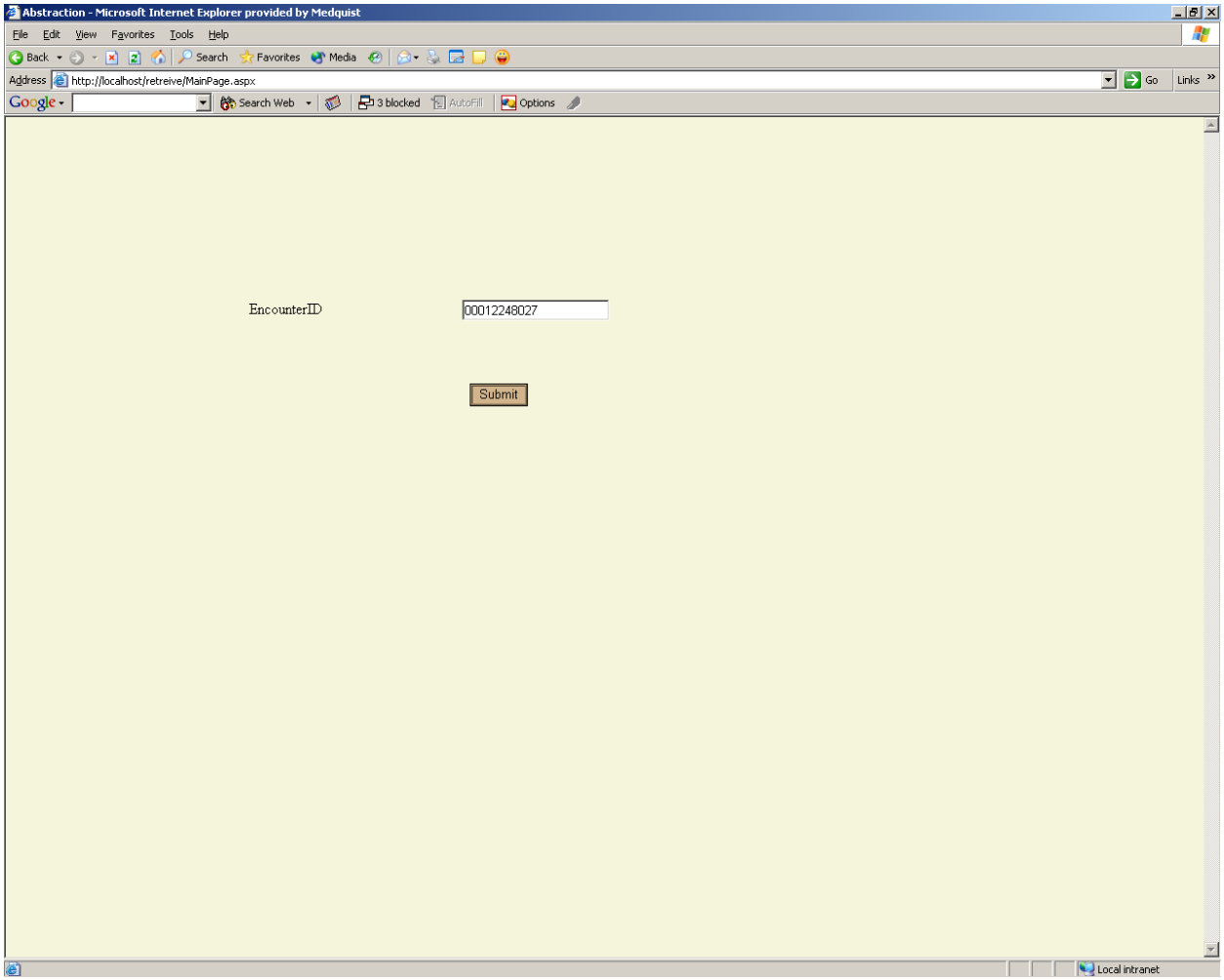
The web interface shown below is the main page of the 'Data Mining Framework'. The web page provides the user three choices viz. Querying Data, Abstracting the quality measures from the XML documents or creating anonymous data (Data Scrubbing). Clicking on either of these buttons, takes the user to web page, which allows him to perform the action of his choice.

Main Page



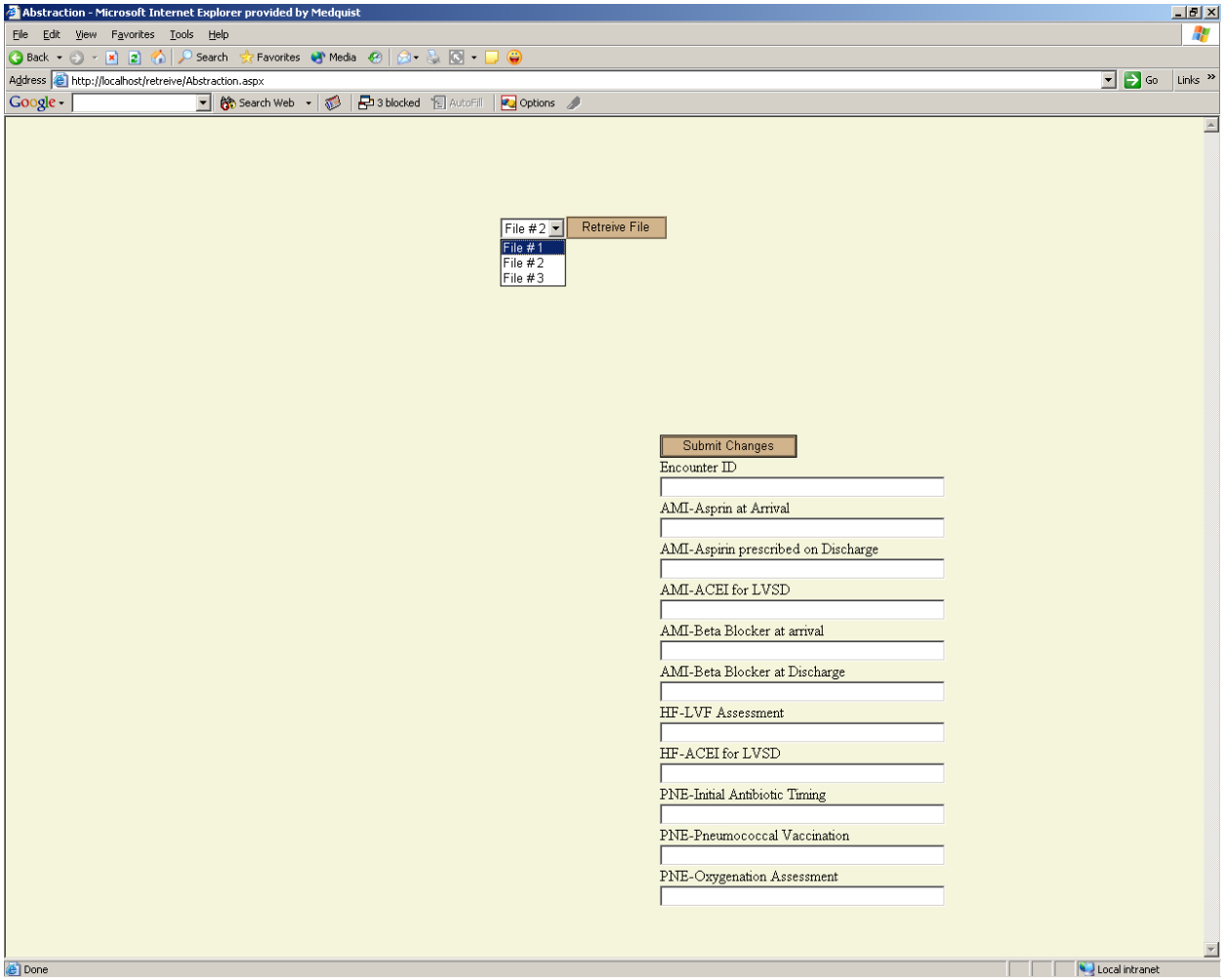
The web page below allows the user to perform manual abstraction of the encounter reports. In order to retrieve all the reports associated with a patient, the encounter ID of the patient is used, as shown below. This ID is used to run a query against the database and retrieve all reports associated with that encounter of the patient.

Data Abstraction



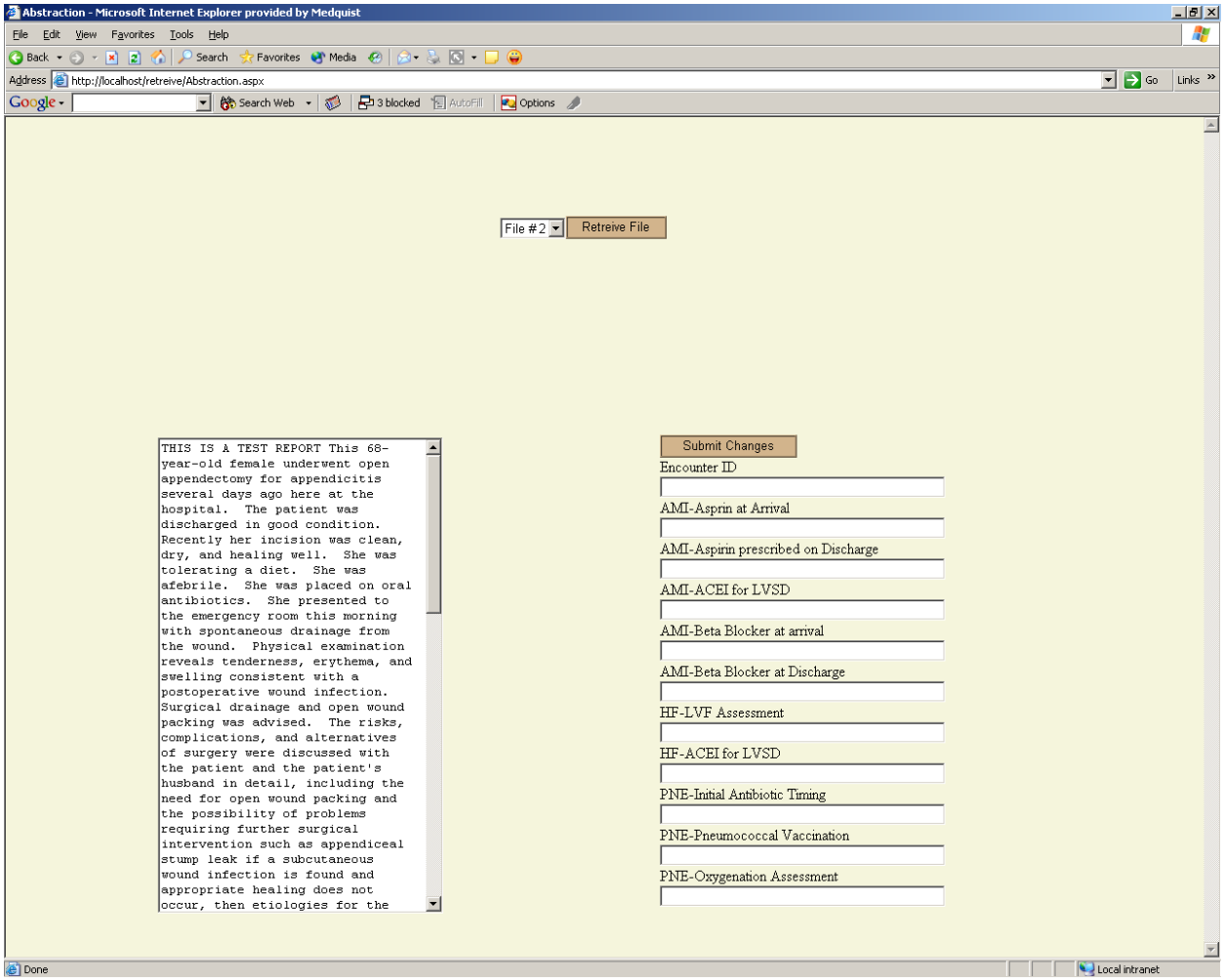
The encounter ID entered in the previous picture retrieved the three files shown below. The user can choose from any of the listed files. The text in the file chosen by the user is displayed when the user hits the 'Retrieve File' button. The user can manually enter the abstracted quality measures into the fields shown below.

Data Abstraction



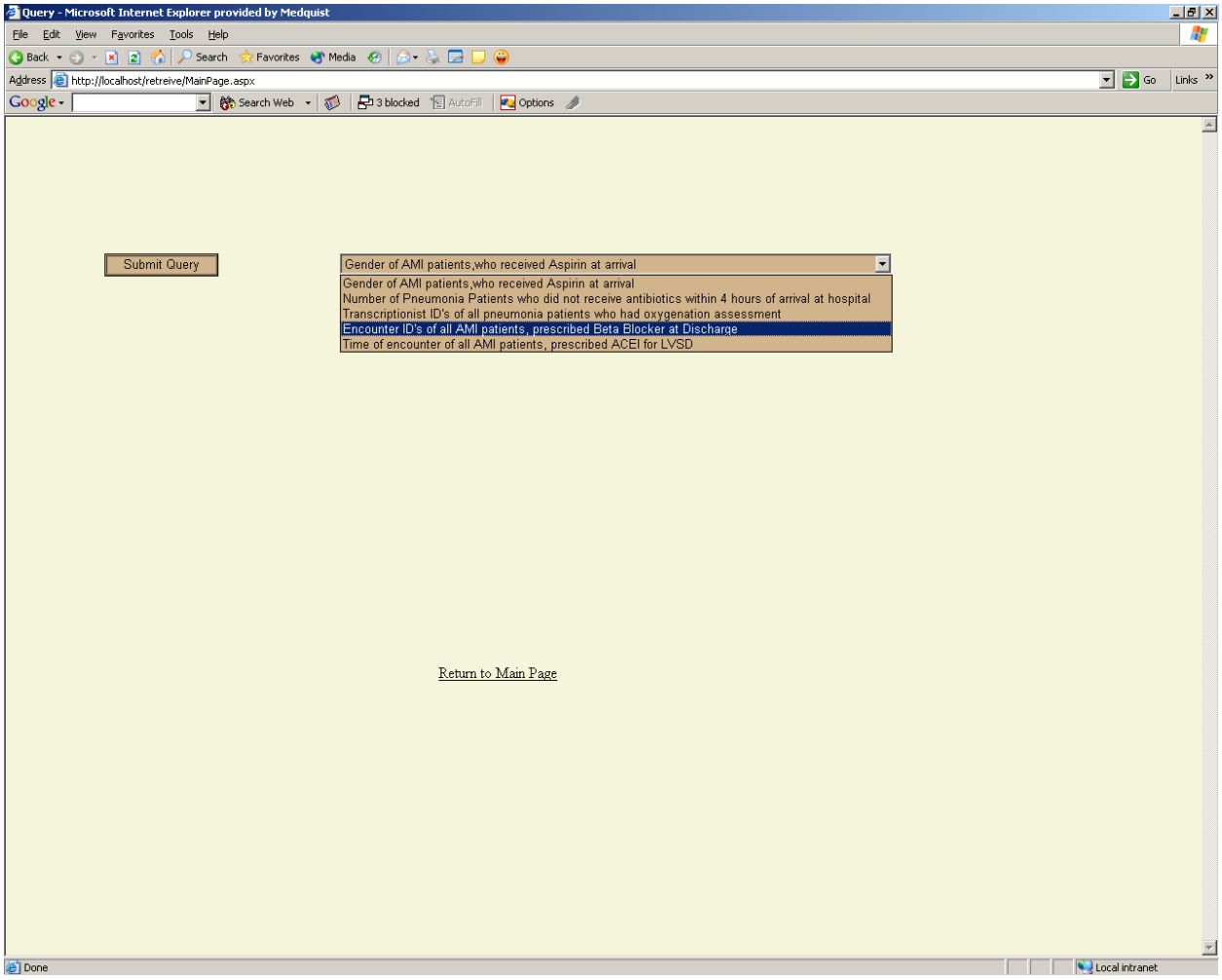
The user chose File#2 to be displayed and hit the ‘Retrieve File’ button. The fields in the element nodes of the body of the CDA document, File#2 are displayed in the text box below. The quality measures can be abstracted from this text box.

Data Abstraction



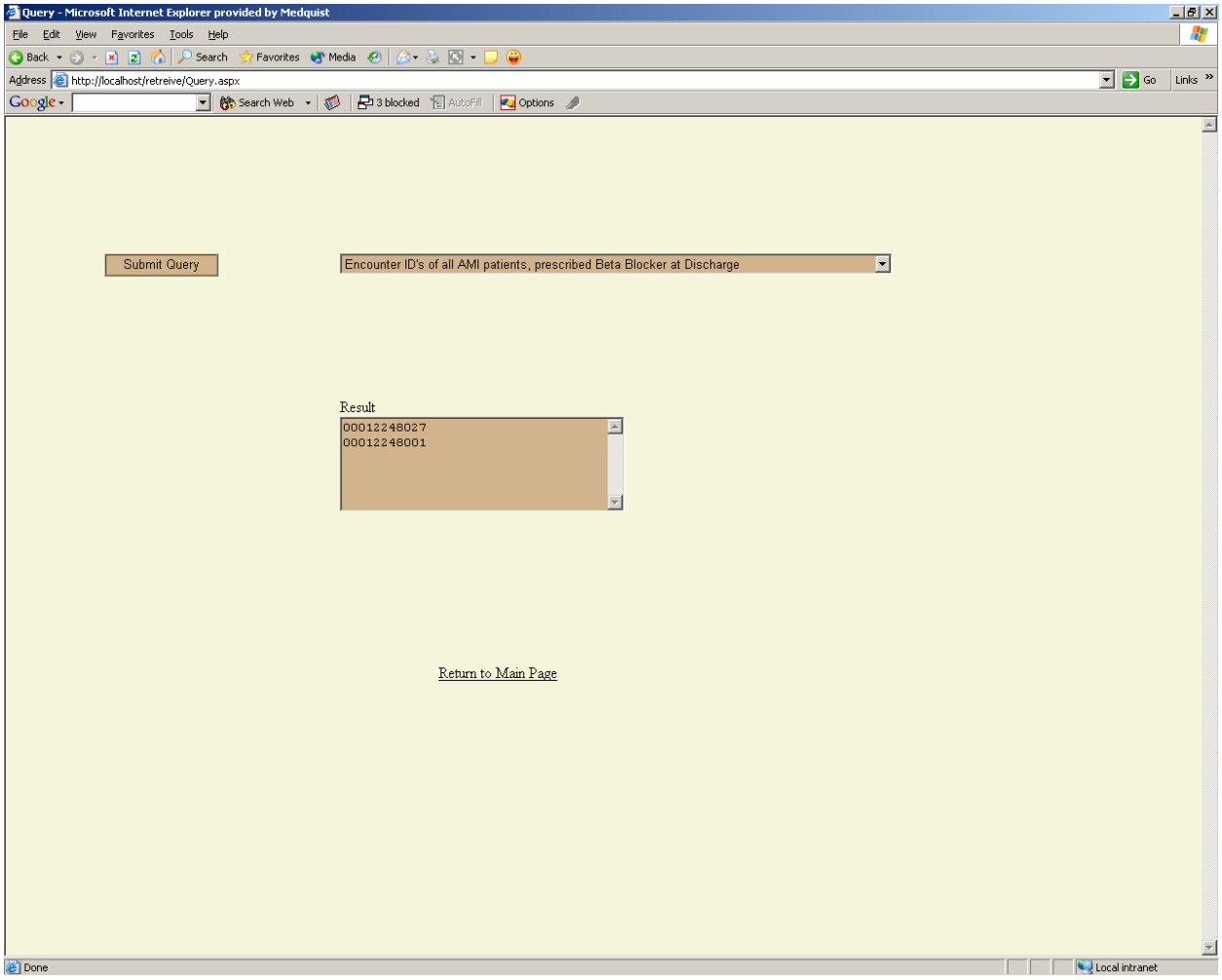
The picture below shows a set of pre-defined queries. The user can chose from any one of the queries listed and hit ‘Submit Query’ to retrieve the results of that query

Data Query



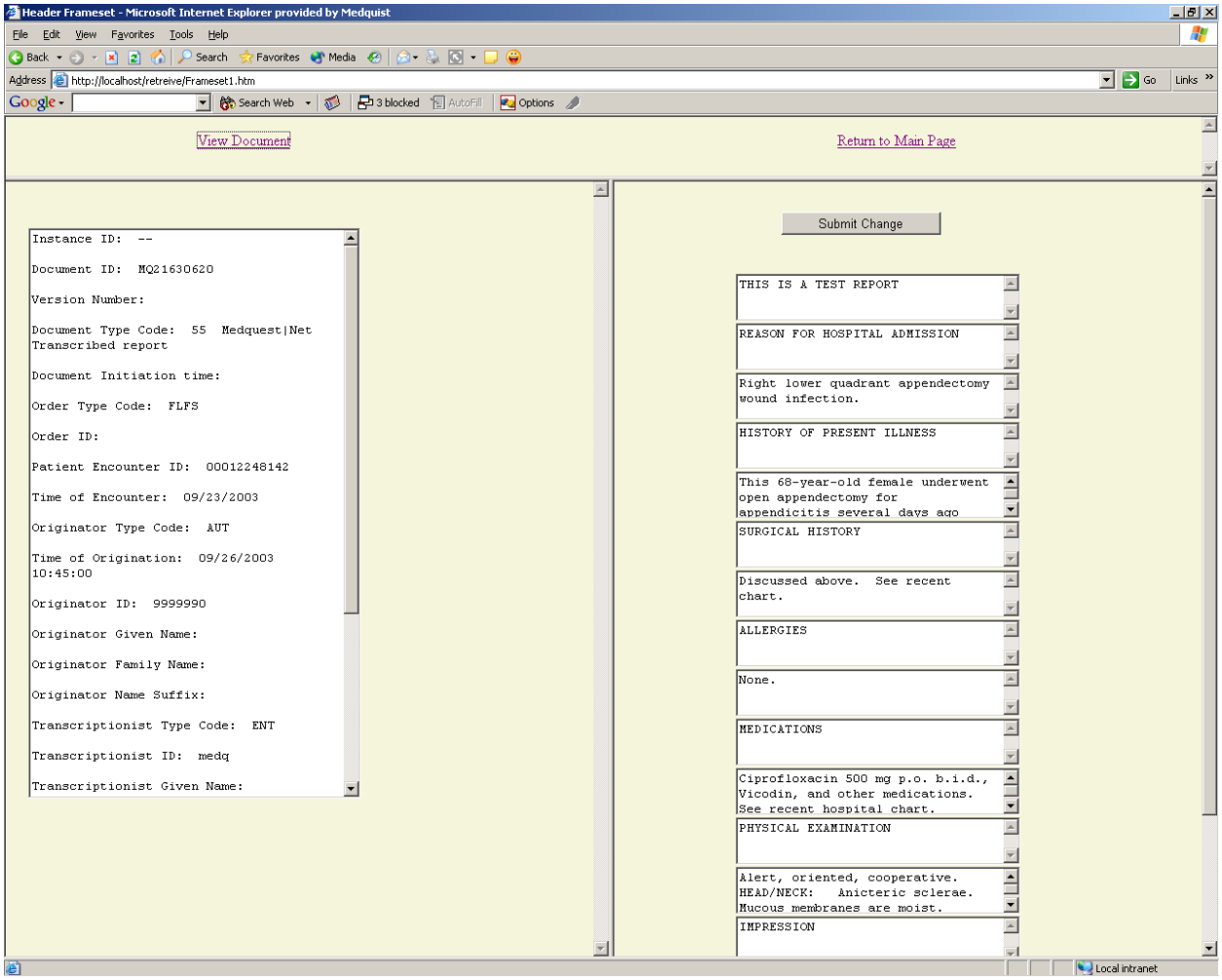
The user chose the query ’Encounter ID’s of all Acute Myocardial Infarction (AMI) patients, who were administered Beta Blocker at discharge.’ The results of that query are displayed in the text box below.

Data Query



The web page below allows the user to create anonymous data. The header and body parts of a CDA document are displayed below. The attribute name and value of the CDA header are displayed in the left frame, while the text in the body of the document are displayed in the editable text boxes in the right frame. Once the changes have been made, the user can hit the 'Submit Change' button to save the results.

Anonymous Data



7.ANALYSIS

This chapter provides an analysis of the working of the Data Mining Framework. Future work that can be done to improve the functioning of the system is also discussed.

7.1 Testing

The Data Mining Framework was subjected to testing, at the completion of each module, since each module could function independently of the other. The following test cases were conducted on the modules

7.1.1 Anonymous Module

Case 1: To test if anonymous patient records are generated and stored into the database

20 CDA Release One documents were stored in a temporary collection named ‘temp’ in TigerLogic and the module was run against this collection. The documents in the ‘temp’ collection were displayed in the web interface in a serial order, starting from the first document. Once the patient information in a document was de-identified, the document was deleted from the temp collection and stored into a pre-defined collection in the database by the module. It was found that masking of patient information performed in the web interface was reflected in the database.

Case 2: To test how the module functions with different versions of CDA documents

A mix of CDA Release One and CDA Release Two documents were stored in the ‘temp’ collection. The module was run against this collection. The documents were de-identified, irrespective of the version of the CDA document. The de-identified documents were stored into separate collections in the database by the module, depending on the version of the CDA document.

Case 3:To test the error recovery of the module

CDA documents that contained sensitive patient information were stored in the ‘temp’ collection Patient information was de-identified in the web interface and the possibility of a failure in the connectivity was simulated. It was found that the document was not deleted from the ‘temp’ collection and was displayed again in the web interface, when the connectivity was established.

7.1.2 Abstraction module

Case 1: To test if all the documents with the relevant encounter id were retrieved by the module

20 CDA documents of which 10 documents had the encounter ID ‘00012248027’ were placed in a collection, against which the module was run. The encounter ID entered in the web interface was ‘00012248027’. All the documents in the collection having that encounter ID were retrieved and were listed in the web interface.

Case 2: To test if the abstracted data is stored as an XML file in the database

CDA documents that were generated, for the encounter ID ‘00012248027’ were abstracted using the module. It was found that the results were saved in XML format by the abstraction module.

Case 3: To test the functioning of the module for different versions of CDA documents

Release One and Release Two CDA documents having the same encounter ID’s were placed in a single collection and the module was run against this collection. It was determined that all documents with the encounter ID entered at the web interface were retrieved, irrespective of the CDA version. Further, the document created, by abstracting these CDA documents was saved into the database in XML form by the module.

Case 4: To test error recovery

An encounter ID, ‘00012248027’, was entered in the web interface. Abstraction of all CDA documents retrieved was performed. A connection failure was simulated and it was found that the partially abstracted document was not saved. Also, it was found that, in addition to retrieving all CDA documents with that encounter ID, the module also retrieved the abstracted document for that encounter ID. This allowed for verification and correction of data by the module.

7.1.3 Query module

Case 1: To test the listed queries

Each of the queries listed on the web interface, was tested, comparing the results generated by the module, against manually extracted results. Each of the queries was found to produce accurate results.

Case 2: To test functionality of the module for different versions of CDA documents

The queries were run against a collection containing different versions of CDA documents. The queries returned accurate values, irrespective of the version of CDA documents.

Each of the modules of the application, when tested individually, produced satisfactory results. After the modules were integrated into the application, testing was performed to ensure, that there were no discrepancies in functionality. The results obtained were satisfactory.

7.2 Improvements

The following changes could be made to the Data Mining Framework in order to improve functionality:

- The abstraction module, as implemented in this application, needs manual intervention. An authorized medical representative identifies the measures that need to be abstracted from the clinical documents, before it is saved into the database. This extraction of information can be made automatic using natural language processing methods.
- Further modules could be added to the existing application to manipulate the clinical data, in addition to the existing query and abstraction modules
- Reporting tools could be added to the Data Mining Framework. For instance, the abstracted quality measures could be reported.
- Means to improve system performance could be studied.
- Means to proof check the CDA documents that are stored in the database, against human errors could be devised, to produce accurate results.

7.3 Conclusion

The main goals of the system viz. providing a means to abstract, query and de-identify patient information have been met. Remote clients, through the Internet or an intranet, can access each of these features as they are provided using web interfaces. Flexibility in the working of the application has been achieved. The application works, regardless of the version of CDA document used to encode the clinical data. The design of the application is such that the type of data abstraction and data query that can be performed on the clinical data can be modified, as per need.

The application was developed using Visual Studio .NET. Hence the machine, on which the application is deployed, needs to have a Windows platform. The database management system employed was Tigerlogic XDMS. This database management system was chosen, because all the clinical information was encoded in XML, in CDA documents.

Each of the modules of the application was tested to analyze the performance of the system. The life cycle model employed in the development of the application was the object-oriented life cycle. The overlap between different phases of the software life cycle allowed a parallelism in the application development. The developed application is easy to use through user- friendly web interfaces.

References

1.Fayyad, U. “Data Mining and Knowledge Discovery: Making Sense Out of Data”. IEEE Expert Intelligent Systems and their Applications, (11):5, 1996, pp.20-25

2.Bresnahan, J. “Health Care Data Mining: A Delicate Operation”. CIO, June 15, 1997, pp.44-54.

3.Dudeck, J. “XML & Health Care”. XML Europe 2000, June 16, 2000

4.Jan Komorowski and Jan Zytkow, editors, Principles of Data Mining and Knowledge Discovery. Proceedings, 1997, pp 68-77

5.Health Insurance Portability and Accountability Act (HIPAA), Indian Health Services (IHS), “ Policies and Procedures for De-Identification of Protected Health Information and Subsequent Re-Identification”

6.Health Insurance Portability and Accountability Act (HIPAA) , Indian Health Services (IHS), “ Policies and Procedures for Use and Disclosure of Protected Health Information for Research Purposes

7.American Heart Association (AHA) “The National Voluntary Hospital Reporting Initiative: A Public Resource on Hospital Performance”

8.George Hripcsak, Gilad J. Kuperman, Carol Friedman, Daniel F. Heitjan, “A Reliability Study for Evaluating Information Extraction from Radiology Reports,” Journal of the American Medical Informatics Association, Volume 6, Number 2, March/April 1999, pp 143-150

9. Marti A. Hearst, “Untangling Text Data Mining”, ACL’ 99 Proceedings

10. J. Shanmugasundaram, K. Tufte, G. He, C. Zhang, D. DeWitt, J. Naughton, “Relational databases for querying XML documents: Limitations and opportunities”, Proceedings of the 25th VLDB Conference, September 1999

11. Jacky W.W.Wan, Gillian Dobbie, “Mining association rules from XML data using Xquery”, ACM International Conference Proceeding Series, Proceedings of the second workshop on Australasian information security, Volume 32, pages 169-174, 2004

12. Robinson, Allen, Comes, Glynn, Greenvoss, Harvey, Nagel, Skinner, Watson," Professional C# 2nd Edition", Wrox Publications, 2002
13. S.R.Schach, "Classical and object-oriented software engineering",McGraw-Hill
14. Liora Alschuler, Robert H.Dolin, Sandy Boyer, Calvin Beebe, Paul V. Biron, Sokolowski, "CDA Framework"
15. Liora Alschuler,Robert H.Dolin, Sandy Boyer, Calvin Beebe, Paul V. Biron, Fred Behlen , "HL7 Clinical Document Architecture Release 2.0 "
16. Greg Gillespie, "Health Data Mangement", Oct 2002
17. Boag, Chamberlin, Fernández, Florescu, Robie, Siméon, "XQuery 1.0: An XML Query Language", Oct 2004