INTEGRATED XML SCHEMA FOR HETEROGENEOUS XML SCHEMAS

HTUN EI EI SAN

M.C.Sc. JUNE 2022

INTEGRATED XML SCHEMA FOR HETEROGENEOUS XML SCHEMAS

By

HTUN EI EI SAN

B.C.Sc.

A dissertation submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science
(M.C.Sc.)
University of Computer Studies, Yangon
JUNE 2022

ACKNOWLEDGEMENTS

First of all, I wish to express my deepest gratitude and sincere appreciation to all persons, who contributed directly or indirectly towards the achievement of completing the thesis and helped me throughout the period of studies in University of Computer Studies, Yangon.

As being one of the first of these persons, I would like to express my appreciation and sincere thanks to **Prof. Dr. Mie Mie Khin**, the Rector of the University of Computer Studies, Yangon, for her kind permission to submit this thesis and general guidance, workable environment during the period of study.

My heartfelt thanks and respect to **Dr. Si Si Mar Win**, Professor of Faculty of Computer Science, University of Computer Studies, Yangon, for her administrative support and invaluable guidance, as Dean of Master Course, throughout the development of the thesis.

I would like to extend gratitude to **Dr. Tin Zar Thaw**, Professor of Faculty of Computer Science, University of Computer Studies, Yangon, for her administrative support and invaluable guidance, as Dean of Master Course, throughout the development of the thesis.

I also deeply thank **Daw Hnin Yee Aung, Lecturer,** Department of English, University of Computer Studies, Yangon, for editing my thesis from the language point of view.

A special word of gratitude is due to my supervisor, **Dr. Thidar Win,** Lecturer, Faculty of Computer Science, the University of Computer Studies, Yangon, for her invaluable recommendations regarding the thesis topic, giving me detailed guidance throughout the work of this thesis and support for my thesis work.

In addition, I would like to thank the board of examiners for making precious comments and detailed corrections to my thesis and those who are pressing power to improve the end result.

Last but not least, I am grateful to my family who has provided fully emotional or physical support throughout my student life. And then, I especially thank all my teachers for their valuable advice, opinions and participation in the seminars. Lastly, I also would like to thank my colleagues, friends and staff of the University of Computer Studies, Yangon for providing necessary information, documentation requirements and collaboration during the seminars.

STATEMENT OF ORIGINALITY

I hereby certify that the work embodies in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

-------------------------	---------------------------
Date	Htun Ei Ei San

ABSTRACT

With the growing popularity of the XML model and the proliferation of online XML documents, the automated matching of XML documents and databases has become a critical problem. Currently, many recent applications are based on XML documents. Schema matching plays a central role in a myriad of XML-based applications. There is an increasing need to develop effective matching systems to identify and discover semantic matches between XML data. XML schema matching methods face several challenges in the form of definition, adoption, use and combination of element similarity measurements. In this system, element type conflicts, constraints, naming conflicts, and the semantic and structural information of two specific XML schemas are solved.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

CHAPTER 1
INTRODUCTION

Nowadays, the popularity of the Internet and the exchange of information have increased the need for shared information formats. XML has created a common, high-quality data format for data exchange and integration between various applications and systems. XML is designed to represent information with the help of tags and to display information in a way that is consistent with the enterprise. XML enables us to design information systems in a natural and recognizable way.

Developers can create their own XML documents that follow certain structural rules. These structural specifications are usually defined by an XML Schema (XSD) and a Document Type Definition (DTD). Due to its advantages, XSD is more widely used than DTD. The XML schema itself is an XML document. It is also more robust than DTD in supporting data types and namespace definitions.

Nowadays, ubiquitous data is expanding greatly, and developers are combining this data to preserve it. Data integration plays an important role in improving the efficiency of information transfer between systems.

However Organizational data represented by different XML systems creates challenging problems when merging data. In a real XML Schema document for applications, many schema elements have the same terms, but they may have different names and structures. On the contrary. Many elements have similar models but different meanings. Therefore, one of the major problems in integrating health data is how to evaluate the similarity of components between XML Schema documents.

There is a lot of research that offers dimensions to measure the similarity of concepts between two documents. However, the majority of them concentrated on determining the name similarity or the structural similarity of the elements in the two documents. Some studies have concentrated on both the name and the structure, but certain factors must be determined by hand using human reasoning. The system provides a fully automated method for determining structural and semantic similarity between elements of two XSD schemes.

1.1 Motivation

Traditional approaches to heterogeneous XML schema integration typically create a comprehensive or minimal schema, but not both. The database structure of a large application is too complex for a single designer to model in a single view. User groups typically operate within an organization and have their own data needs and expectations that may conflict with other user groups. Two challenges lead to structural and semantic conflicts in pattern heterogeneity. First, a structural conflict occurs when the same relationship is represented by another XML structure. Second, semantic conflicts also arise when different sources describe the same concept with different element names, or when there is overlapping meaning between similar concepts from different sources.

1.2 Related Works

Many researchers do research on element similarity in xml schema matching and on clustering XML documents.

Husam Ahmed Al Hamad [6] introduced a new technology to integrate different eXtensible Markup Language (XML) systems, under the name XDEHD. A shared schema that has all the ideas and connections from the source without duplication. The system is divided into three steps: First, separate schema to extract all sub-schemas; each sub-schema consists of three stages: ancestor, root and leaf. After that, the technology matches and compares the sub- schema to convey related sub-schema candidates, and the semantic closeness function is used to determine how similar the sub-schema terms are formatted in the source. This system aims to create a comprehensive shared schema between heterogeneous database sets by combining XML system resources.

Xia Yang, Mong Li Lee, Tok Wang Ling [15] have indeed implemented a semantic approach to resolve structural conflicts in the integration of XML schemas. The system uses a data model called ORA-SS (Object Relation Attribute Model for Semi-Structured Data) to capture the semantics contained in the XML schema. Provides a complete algorithm for integrating XML schemas. Compared to other methods, this system uses an n-nary integration strategy that takes into account the semantics of the data, the importance of the source, and how most sources sample the data for structural conflicts such as attribute / object class conflict and ancient dissolution. conflict. This system resolved structural conflicts such as attribute / object class conflicts, decreasing conflicts. The proposed technique has primarily resolved structural conflicts, but most semantic conflicts have not been resolved.

May Myat Thu [12] uses the XEdge algorithm to classify XML files. XML files are classified using the Edge representation in the XEdge group mask. XML files are inserted first, and then a tree structure is created. The tree node symbol is now stored in memory. If more data is generated, the node ID will be checked. Next, the Stage Structure representatives of the input XML files are created. This can result in all results being in a uniform XML file with the same set of tags, or in the case of XML files, with the same set of tags. Sharing XML files is useful for XML applications such as XML search.

1.3 Objectives of the Thesis

The main objectives of the thesis are as follows:

- To provide users with easy and simple information
- To offer developers with a systematically integrated schema.
- To produce an integrated schema with accurate and minimal: structure and semantic.
- To propose a semantic and structure similarity approach for XML Schemas.
- To compute the element types and measure the cardinality constraints of two elements.

1.4 Organization of the Thesis

This thesis is mainly composed of five chapters.

Chapter 1 is the introductory section where the introduction of integrated XML Schema Based on Heterogenous XML Schemas, the related works, the objectives and the organization of the thesis are presented.

Chapter 2 describes the background theory related to this thesis such as schema integration techniques, integration processing strategies, XML data model, XML Schema (XSDs).

Chapter 3 presents the design of the proposed system that is described as the system flow, description about electronic healthcare record EHR XML dataset that is used, the detail steps of Semantic Similarity Measurement, Element Type Similarity, Constraint Similarity, Linguistic Similarity Algorithm, Structure Similarity Measurement, Structure Similarity Algorithm.

Chapter 4 primarily explains the proposed system's implementation in detail, including the experimental setup, system implementation, healthcare record EHR XML dataset process, and experimental result.

Finally, Chapter 5 concludes this thesis by highlighting the proposed system's limitations and future improvement efforts.

CHAPTER 2

BACKGROUND THEORY

The goal of schema integration is to create a mediatized schema as a unified representation of existing heterogeneous sources that share a common function. Because of the versatility and format of these sources, they are mostly written in XML.

2.1 Schema Integration

Schema integration is the consolidation of existing data source schemas into a single schema, called a global schema or integrated schema. This federated schema serves as a unified interface for querying data sources. However, the integrated schema can be used for many other applications. In fact, due to the increasing availability of information about companies, institutions, or the Internet, policymakers need to quickly understand some concepts, such as creating communities of interest before taking action. There are two types of XML integration:

2.1.1 View Integration

The goal is to create an integrated schema through a self-contained application concept collection. For large systems, the database system is too complex for a single manufacturer to copy in one view. In many companies, teams work independently and have the data they need and expectations that may go against the demands of other employees.

2.1.2 Database Integration

A distributed database is a collection of data that logically belongs to the same system but is distributed across multiple points in a computer network. Database integration creates a single schema for the database group. A global schema is a virtual representation of all databases in a distributed database management system.

2.2 Integration Processing Strategies

Integrated strategies are processes that an organization can use to increase its competitiveness, efficiency or market share by increasing its exposure to a new location. These venues can supply, offer or compete. Each division has different rules of integration and there are many methods that the organization can use.

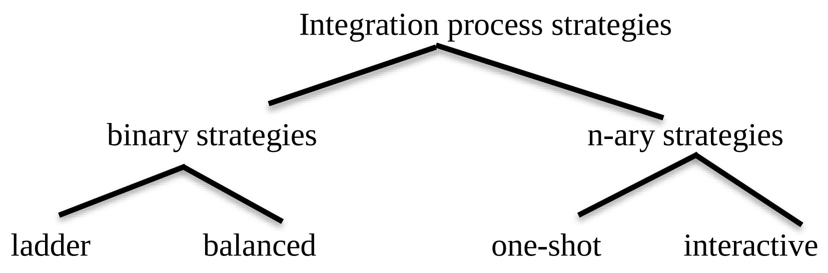


Figure 2.1 Integration Processing Strategies

2.2.1 Integration Process

(1) Pre-Integration

- Select strategies for the integration process
- This governs the selection of integration plans

(2) Compare the Schemas

- Analyze strategies and comparisons to determine communication between concepts and identify conflicts.

(3) Conform the Schemas

- Once conflicts have been identified, we seek to resolve them so that the different patterns can be brought together.
- Automatic conflict resolution is usually not feasible; communication with designers is essential.

(4) Merging and Restructuring

- Schemas are ready to overlap, resulting in some integrated intermediate schemas.

2.3 XML Data Model

The information demonstrates for XML is exceptionally basic or exceptionally theoretical, depending on one's point of view. XML gives no more than a pattern on which more complex models can be built. The reason of the information show is to characterize all passable values of expressions in XPath, counting values that are utilized in the middle of the road calculations. Each XPath expression takes as its input an occurrence of the information demonstrates and returns an occasion of the information show.

(1) Sequences and Items

The XPath data model is based on the notion of a sequence. The value of an XPath expression is always a sequence. A sequence is an ordered collection of zero or more items. An item is either an atomic value or a node.

(2) Atomic Values

An atomic value is an instance of one of the built-in atomic data types that are defined by XML Schema.

(3) Nodes

A node conforms to one of the types of nodes that are defined for XPath. These node types include document, element, attribute, text, processing instruction, comment, and namespace nodes.

(4) Data Model Generation

Before an XPath expression can be processed, the input documents must be represented in the XML data model.

2.3.1 Schemas and XML Data Modeling

The process of creating a schema for an XML document is also known as building a data structure because it involves parsing data classes and references into reference data classes that can be used to display information in the XML document. The real importance of schemas is that they legitimize the accuracy of XML documents. The main reason to use schemas in XML is to enable machine document validation. Simply put, a schema allows an XML developer (or application) to execute the document and obey any restrictions specified in the schema. A valid XML document acts like a validation stamp, declaring the appropriate document for use in an XML application.

```xml
<WomenClothing>
<TopsSets>
<TankTops> </TankTops>
<T-Shirts> </T-Shirts>
<Polo> </Polo>
<Sets> </Sets>
<Jumpsuits> </Jumpsuits>
<Rompers> </Rompers>
</TopsSets>
<Bottoms>
<Jeans> </Jeans>
<PantsandCapris> </PantsandCapris>
<Shorts> </Shorts>
<SocksHosiery> </SocksHosiery>
</Bottoms>
<OuterwearJackets>
<Basic-Outerwear> </Basic-Outerwear>
<HoodiesandSweatshirts> </HoodiesandSweatshirts>
<Blazers> </Blazers>
</OuterwearJackets>
<Weddings-Events>
<Dresses> </Dresses>
<Wedding-Dresses> </Wedding-Dresses>
<Evening-Gowns> </Evening-Gowns>
</Weddings-Events>
<TraditionalandCeremonialClothing>
<Sets> </Sets>
<Tops> </Sets>
<Longies> </Longies>
</TraditionalandCeremonialClothing>
<Accessories>
<Belts> </Belts>
<ScarvesandWraps> </ScarvesandWraps>
<HatsandCaps> </HatsandCaps>
<HairAccessories> </HairAccessories>
<TiesandHandkerchiefs> </TiesandHandkerchiefs>
<Masks> </Masks>
</Accessories>
</WomenClothing>
```

Figure 2.2 Example of XML Document

The application developer must write a separate code to ensure that certain namespaces are complied with, such as email addresses separated by a "at" symbol in the name and domain name. The creator of the XML document uses the schema while the email application software developer is coding to verify the authenticity of an email address. XML applications can be used to ensure that documents are legitimate. Schemas provides a mechanism to streamline the process of validating XML documents.

When it comes to creating schemas, there are two primary approaches:

- Document Type Definitions (DTDs)
- XML Schemas (XSDs)

2.3.1.1 Document Type Definitions (DTDs)

DTD, which stands for Document Type Definition. DTDs represent the original method for creating schemas for XML documents. DTDs are not derived from XML. DTDs have their origin in XML's predecessor, SGML (Standard Generalized Markup Language). The main drawback to DTDs is that they are based upon a somewhat cryptic language. XML provides a highly structured approach to formatting data.

Here's an example of a DTD that could store a list of basketball players on a team:

1. < !ELEMENT player_list (player) *>
2. < !ELEMENT player (name, age, school? , country)>
3. < !ELEMENT name (#PCDATA) >
4. < !ELEMENT age (#PCDATA) >
5. < !ELEMENT school (#PCDATA) >
6. < !ELEMENT country (#PCDATA) >

The first line says that player_list is a valid element name and that any instance of that element contains any number of player elements. * Indicates that there may be 0 or more player elements in the player_list element. The next line declares that player is a valid element and that any instance of that element must be immediately followed by an element of type name, then age, then school (optional), and finally country. East ? Characters following an element indicate that the element is optional. The third, fourth, fifth, and sixth lines only declare the item name, age, school, and country as valid item types. The (#PCDATA) tag represents parsed character data, meaning that the data is taken from what the document author entered.

2.3.1.2 XML Schema (XSDs)

The XML Schema replaces DTDs in a more robust and intuitive way for XML-based markup languages. Schemas created using the XML Schema are coded in the XSD (Definition of XML Schema) language, hence the name XSD. The XML Schema and XSD languages were created by the W3C (World Wide Web Consortium) and represent a more robust and flexible approach to schemas than DTDs. The idea behind the XML Schema is to use XML as a basis for creating schemas.

An XSD is very similar in purpose to a DTD in that it is used to create a schema for a class of XML documents. Like DTDs, XSDs describe elements and their content model so that documents can be authenticated. However, XSD takes a few steps beyond DTDs by allowing data types to be linked to elements. In a DTD, the content of an element is largely restricted to text. XSD is more flexible, you can set the data type of an element to a specific type, such as an integer or a date. Here is a sample XML schema for patient.

```
<xs:schema targetNamespace="org.di.demo.patient" xmlns="org.di.demo.patient"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="PatientList">
<xs:complexType>
<xs:sequence>
<xs:element maxOccurs="unbounded" minOccurs="0" name="Patient"
type="PatientElement"/>
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:complexType name="PatientElement">
<xs:sequence>
<xs:element minOccurs="1" name="firstName" type="xs:string"/>
<xs:element minOccurs="1" name="lasttName" type="xs:string"/>
<xs:element minOccurs="1" name="middleName" type="xs:string"/>
<xs:element minOccurs="1" name="ssn">
</xs:element>
<xs:element minOccurs="1" name="sex" type="xs:string"/>
<xs:element minOccurs="1" name="dob" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:schema>
```

Figure 2.3 Example of XML Schema

2.3.1.3 Why XML is important?

In the mid-1990s. Extensible Markup Language (XML) has become widely accepted as an exchange framework due to the growing demand for a common platform that enhances interoperability between companies. Today Most data exchanges are based on XML-based data representation standards. XML provides simple, flexible, self-representation of your data. Its flexibility is due to the fact that segregation can be used to effectively integrate other alternative forms. In addition, XML events are self-explanatory because they hold the data structure in the form of human-readable tags associated with the data element. Therefore, XML data can be exchanged without a sequence. This simplicity and compatibility make it possible to use XML on many different domains, a key requirement for easy data exchange.

2.4 Semantic Matching

Many well-known metadata-intensive applications, such as schema/ontology integration, data warehouses, data integration, e-commerce, and so on, rely on matching. The match operator takes two graph-like structures and generates a mapping between the graph nodes that correspond semantically. The system focuses on a schema-based solution, specifically a matching system that only uses the schema information. The semantic matching approach is founded on two key concepts:

- The concept of a label denotes the set of documents (data instances) that would be classified under the label it encodes.
- Concept at a node, which denotes the set of documents (data instances) that would be classified under a node if it had a specific label and was located in a specific position in a tree.

Semantic matching approach computes the concepts of labels according to the following four logical phases.

- Tokenization. Labels of nodes are parsed into tokens by a tokenizer which recognizes punctuation, cases, digits, stop characters, etc.
- Lemmatization. Tokens of labels are further lemmatized, namely they are morphologically analyzed in order to find all their possible basic forms.
- Building atomic concepts. WordNet is queried to obtain the senses of lemmas identified during the previous phase.
- Building complex concepts. When existing, all tokens that are prepositions, punctuation marks, conjunctions are translated into logical connectives and used to build complex concepts out of the atomic concepts constructed in the previous phase.

2.5 Data Integration

Data integration is the process of combining numerous disparate and independent data sources. Its goal is to provide users who need to search or analyze multiple data sources with a logically unified view of data. In the data management community, data integration is a well-studied problem. Nonetheless, despite decades of work in the field, issues persist. The system focuses on techniques for integrating Extensible Markup Language (XML) data. XML offers the opportunity to improve data source compatibility. XML also presents novel challenges that necessitate creative solutions.

Many applications need to exchange and combine data from many different sources. Extensible Markup Language (XML) is a standard developed to meet the needs of these applications to facilitate data exchange. XML is a standard that companies can use to define and implement transparent integration capabilities. XML-oriented features provide tremendous value to an organization's information architecture, but it doesn't just apply to an organization's IT functions. XML data integration provides a clear definition of organization and terminology, closely related to key management initiatives. Powerful as a business intermediary with an emphasis on disclosure, the organization's XML data integration capabilities and applications represent a thought-provoking and thought-provoking business asset and ensure strategic business oversight.

CHAPTER 3
DESIGN OF THE PROPOSED SYSTEM

This system will present generating a complete and minimal global schema based on heterogenous XML schemas. With a goal of creating a complete and minimal schema, this integration approach solves through two main conflicts: (i) semantic conflicts and (ii) structure conflicts.

3.1 Process Flow of the Proposed System

The proposed methods in this system present a new technique for a comprehensive global schema that effectively integrates two heterogenous XML schemas. The system gives the integration of relations over that of separate concepts because the relations carry domain information. In addition to having similar characteristics, the solution has the following properties:

- It automatically enables basic language compatibility.
- It is elementary and structural.
- It is one-sided towards the uniformity of the leaf elements.
- It uses an internal structure.

Figure 3.1 shows the overview of the proposed system. The proposed system presents semantic and structure measurement to generate an integrated schema. In order to the system flowchart, there are four main steps in the system. Firstly, it accepts two heterogenous XSD sources and decomposes them into subschemas from all sources. Each subschema has parent and its child elements. The second step computes semantic between elements from all sources using three processes. For the previous semantic step, three phases are considered: first the system finds element similarity which is complex of simple, the Occurrences of elements in sources and final phase is used to compute naming similarity using WordNet. The next step is to find structure similarity between element pairs that it matches the schema elements based on the similarity of their position and their nearest elements. The final step is to generate an integrated XML Schema.

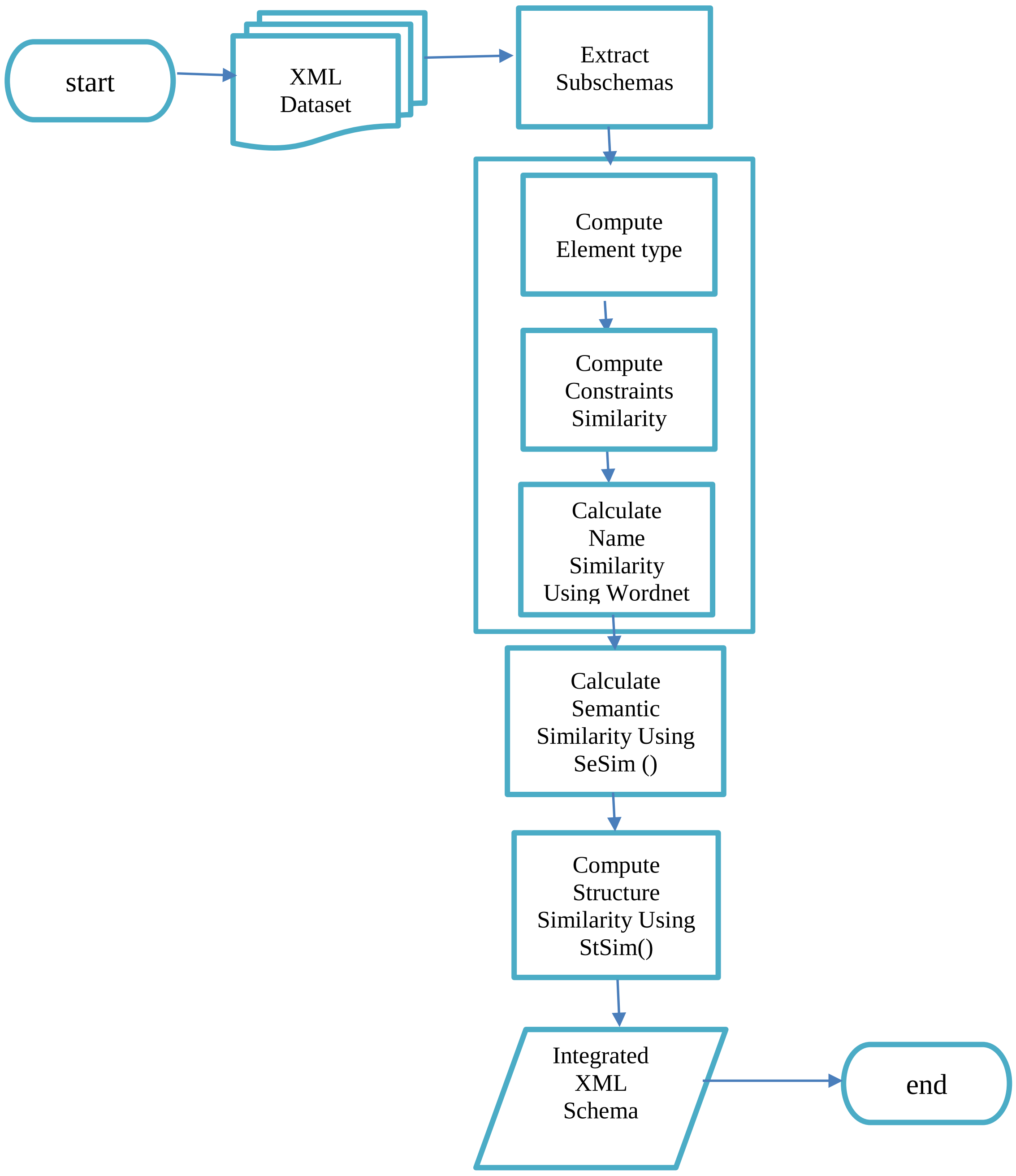


Figure 3.1 Overview of the Proposed System

3.2 Overview of XML Schema Dataset

In this system, electronic healthcare record EHR XML datasets are used as input files. The XML schema datasets of healthcare are from https://data.world/healthcare. The following XML schemas are sample for input files of the proposed system.

```xml
<?xml version="1.0" encoding="utf-8"?>
<!-- Created with Liquid Technologies Online Tools 1.0 (https://www.liquid-technologies.com) -->
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="patient">
<xs:complexType>
<xs:sequence>
<xs:element name="person.name">
<xs:complexType>
<xs:sequence>
<xs:element name="firstname" />
<xs:element name="lastname" />
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="id">
<xs:complexType>
<xs:sequence>
<xs:element name="type" />
<xs:element name="authority" />
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name="address">
<xs:complexType>
<xs:sequence>
<xs:element name="street" />
<xs:element name="city" />
<xs:element name="country" />
<xs:element name="postcode" />
</xs:sequence>
</xs:complexType>
```

```
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
```

Figure 3.2 Sample XML Schema File 1

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Created with Liquid Technologies Online Tools 1.0 (https://www.liquid-technologies.com) -->
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="patient">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="name">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="given" />
              <xs:element name="family" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="id" />
        <xs:element name="address">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="street" />
              <xs:element name="city" />
              <xs:element name="country" />
              <xs:element name="housenumber" />
            </xs:sequence>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
</xs:schema>
```

Figure 3.3 Sample XML Schema file 2

The semantics of concepts play an important role in the integration of text documents. XML Schema Semantics includes dictionaries, content models, and data types. Typically, the XML Schema uses a standard namespace (xs or xsd) and a URI associated with that namespace to launch a document. Using the XML Schema, the system can determine the possible number of occurrences of an element using the maxOccurs and minOccurs attributes. Moreover, simpleType or complexType elements help us identify the similarity of data types between two elements.Because the element names of two elements differ, this system must compute their linguistic similarity using WordNet. Person and name are tokenized from the element person.name. In Figure 3.5, the final token corresponds to an element name. As a result, the linguistic similarity between person.name and name is 0.8.

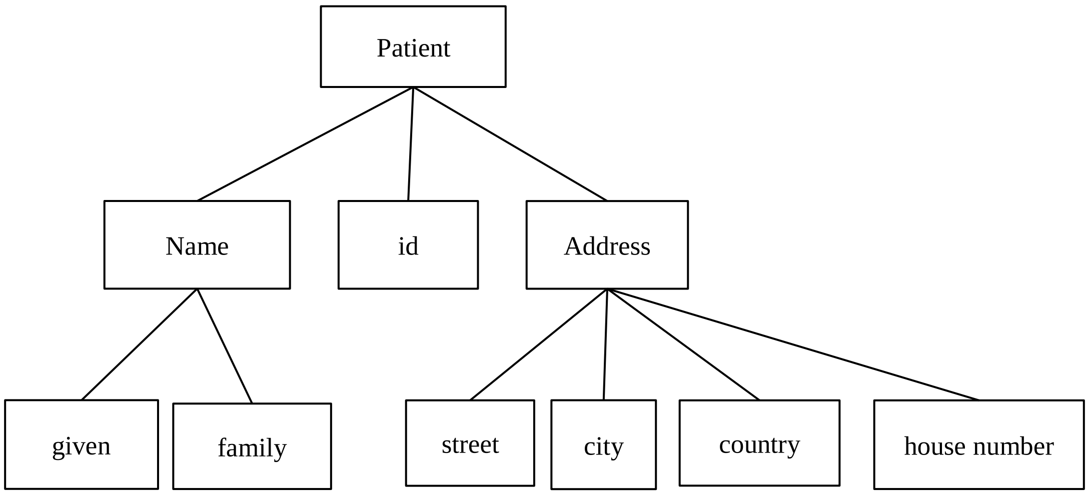


Figure 3.4 Tree of XML Source 1

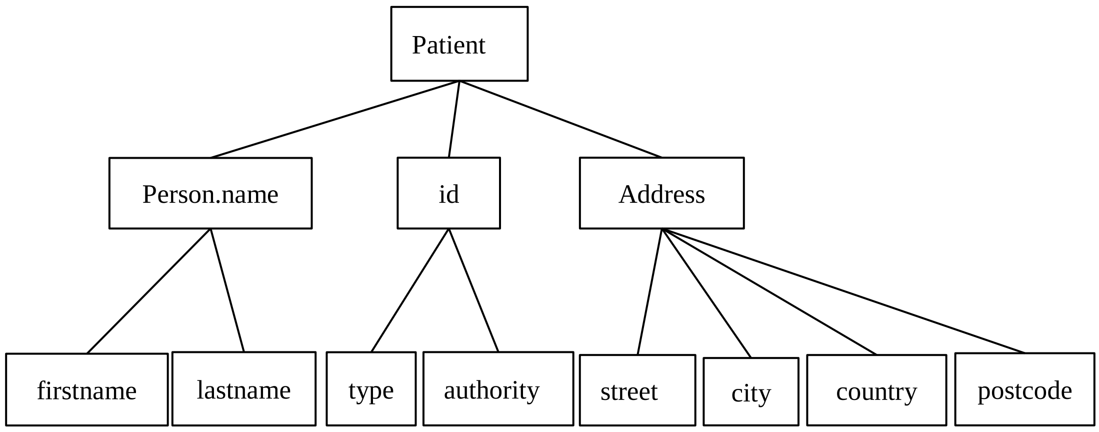


Figure 3.5 Tree of XML Source 2

3.3 Semantic Similarity Measurement

The semantics of concepts are crucial in the integration of text documents. Dictionaries, content models, and data types are all part of XML Schema Semantics. To launch a document, the XML Schema typically uses a standard namespace (xs or xsd) and a URI associated with that namespace. The system can use the XML Schema to determine the maximum and minimum number of occurrences of an element by using the maxOccurs and minOccurs attributes. Furthermore, simpleType and complexType elements aid in determining the data type similarity between two elements.

3.3.1 Element Type Similarity

While the name of the element is the most important factor in calculating semantic similarity, other components must also be considered. The two id elements in Figures 3.4 and 3.5, for example, have a name similarity of 1. However, this is an incorrect value because the first id element is complex and the second id element is single. This means that they are distinct from other traits. As a result, other factors must be used to calculate their semantic relationship in order to eliminate some incorrect matches. Each element in an XML Schema document is either a simple or complex type. If two elements have the same name and have the same child elements, their semantic similarity may be greater than in other cases, both simple and complex. Because complex elements contain child elements, the system must compare the similarity of their child elements to calculate the similarity of two complex elements.

3.3.2 Constraint Similarity

The cardinality (occurrence) constraint is another factor that influences the semantic similarity of two elements. The minOccurs and maxOccurs attributes define the minimum and maximum number of times an element appears in XML instances. The system specifies the constraint similarity between two elements e1 and e2 using CSim (e1, e2). In contrast to the proposed constraint table, where values are determined by human judgment. The following equation is used to compute cardinality constraint similarity for the values of minOccurs and maxOccurs:

$$CSim(e1(min,,\mathrm{max}),e2(min,\mathrm{max}) = \frac{\left(1-\frac{|\mathrm{e1.min}-\mathrm{e2.min}|}{\mathrm{e1.min}+\mathrm{e2.min}}\right)+1-\frac{|\mathrm{e1.max}-\mathrm{e2.max}|}{\mathrm{e1.max}+\mathrm{e2.max}}}{2} \qquad \textbf{(3.1)}$$

In the above equation, min and max are short forms of minOccurs and maxOccurs, respectively.

	Min=0 Max=unbound	**Min=1 Max=unbound**	**Min=0 Max=1**	**Min=1 Max=1**
min=0, max=unbound	1.00	0.5	0.67	0.17
min=1, max=unbound	0.5	1.00	0.17	0.67
min=0, max=1	0.67	0.17	1.00	0.5
min= 1, max= 1	0.17	0.67	0.5	1.00

Table 3.1 Constraint Similarity

3.3.3 Name Similarity

The linguistic similarity of the two elements is the most important factor for semantic measurement. The system employs the algorithm presented in the following language similarity algorithm to determine the linguistic similarity between elements. In fact, the algorithm finds similarities between two components e1 and e2. Scope: The first search will be performed on WordNet from the E1 element sensor to the E2 element sensor and so on until it is synchronized with e2. Language Sync returns a value of 0 if no target is found, otherwise it is calculated as a distance of 0.9.

```
Algorithm : Name Similarity
Input: Two elements, e1 and e2
Distance = 5-level;
Output: Name similarity
If e1.name==e2.name then return 1;
else return DepthSyn (e1, {e2}, level);
{e2} =S;
Function DepthSyn (e, S, level)
Output: the synonym in depth
If (level>=distance) then return 0;
Else if (e1∈S) then return power (0.9, distance);
Return DepthSyn (e, S, distance+1);
```

Figure 3.6 Linguistic Similarity Algorithm

Definition 1: Semantic similarity captures the similarity between two elements' names, constraints, and path context. This is provided by:

$$SeSim(e1, e2) = \propto * NameSim(e1, e2) + \beta * EleSim(e1, e2) + (1 - \alpha - \beta) * CSim(e1, e2) \quad \textbf{(3.2)}$$

Where SeSim is the semantic similarity and a and b are the program's weighted constants. NameSim is the name similarity calculated by the previous Linguistics Similarity Algorithm algorithm. EleSim denotes the similarity of two element types, whereas CSim denotes the cardinality constraint similarity of e1 and e2 elements.

The similarity score between two components using Wordnet is displayed in table 3.1 below. First, from level 0 to level 2, there are 16 separate pieces spread between two systems. The system will match each element in source 2 with element 1 of source 1, as shown in Figure 3.4. If two elements match, the Linguistics Similarity Algorithm assigns a similarity value of 1 to them. The second aspect is a person-to-person similarity. both names. This system must use WordNet to determine the linguistic similarity of two elements because their names differ. Person and name are tokenized from the element person.name. The element name in Figure 3.5 matches the last token. As a result, person.name and name have a 0.8 linguistic similarity.

Element1 of Source1	**Element2 of Source2**	**Name Similarity Using Wordnet**	**Level**
Patient	Patient	1	0
Person.name	name	0.8	1
firstname	given	0.8	2
lastname	family	0.8	2
id	Id	1	1
type		0	2
authority		0	2
address	address	1	1
street	street	1	2
city	City	1	2
country	country	1	2
postcode		0	2

Table 3.2 Name Similarity

3.4 Structure Similarity Measurement

The second stage is referred to as structure matching. It matches schema elements based on the similarity of their context (position) and the elements closest to them. The structure matching is partially determined by the semantic similarity computed in the first stage. The result is a structure similarity coefficient, StSim, for each pair of elements. If two elements are similar in contexts, they have structural similarity. The structure similarity algorithm computes structure similarity using the following principles:

- If the elements in their ancestors and siblings are identical, as well as if their tags are, then the elements that are leaves of the two trees are comparable.
- If the sub-tree rooted at two non-leaf items is similar and their tags are similar, then the two elements are similar.
- And even if their immediate children are not, two non-leaf elements are structurally similar if their leaf sets are quite similar.

```
Algorithm: Structure Similarity:
Input: Two schema trees S,T
Thresh_min=0; thresh_max=0.3
Output: The structure similarity
For each s ϵ S, t ϵ T where s, t are leaves
S1=post-order(S); S2=post-order(T);
for each s in S1,
    for each t in S2 tϵT
        if StSim (s,t)>=thresh_max then
            StSim(s,t) = StSim(s,t)+0.1;
        else if StSim (s,t)<=thresh_min then
            StSim(s,t) = StSim(s,t)+0.1;
Structure_Similarity(S,T)= StSim(s,t);
```

Figure 3.7 Structure Similarity Algorithm

Definition 2: The structure similarity between two elements e1 and e2 is specified as:

$$\mathrm{StSim}(e1, e2) = \frac{\mathrm{sum\_links}(e1,e2)+\mathrm{sum\_links}(e2,e1)}{\mathrm{leaves}(e1)+\mathrm{leaves}(e2)} \quad \textbf{(3.3)}$$

Where leaves (e1) is the total number of leaves in the subtree rooted at element e1 and sum links (e1, e2) is the total number of links from element e1's leaves to element e2's leaves. Following the matching, the information values of all source leaves are quickly organized as candidate subschemas. Because the structure of a document is associated with the semantic similarity of its elements, the system assumes that condition. If semantic similarity is not taken into account, it may result in two schema trees having the same structure. For example, compute the structural similarity of two elements, as shown in Figures 3.4 and 3.5.

- leaves(address) + leaves (address!) = 4 + 4
- sum_link (address, address!) = 3
- sum_link (address! , address) =3
- StSSim(address, address!) = 6/8 = 0.75

Element1 of Source1	**Element2 of Source2**	**Structure Similarity Using StSim(e1,e2)**	**Level**
Patient	Patient	1.1	0
Person.name	Name	0.9	1
firstname	Given	0.9	2
lastname	Family	0.9	2
Id	Id	1	1
Type		0	2
authority		0	2
address	Address	0.85	1
Street	Street	1	2
City	City	1	2
country	Country	1	2
postcode		0	2

Table 3.3 Structure Similarity

3.5 Create an Integrated XML Schema

The system considers both the structure and semantics of the schemas. As a result, in both XML Schema graphs, the similarity between two elements is calculated as the weighted sum of these two components.

$$\mathrm{ESim}(s,t) = \delta * \mathrm{SeSim}(s,t) + (1-\delta) * \mathrm{StSim}(s,t) \qquad \textbf{(3.4)}$$

where $\boldsymbol{\delta}$ is the weighted value, $0 < \boldsymbol{\delta} <= 1$.

To understand the system, compute the element similarity of some pairs of elements given in Figure 3.4 and Figure 3.5. The system distinguishes between elements with the name labels in two schemas, for instance, it computes the similarity between two elements using ESim method. This system defines how elements can be semantically and structurally equivalent, and then produces an integrated schema based on semantic and structure measurement.

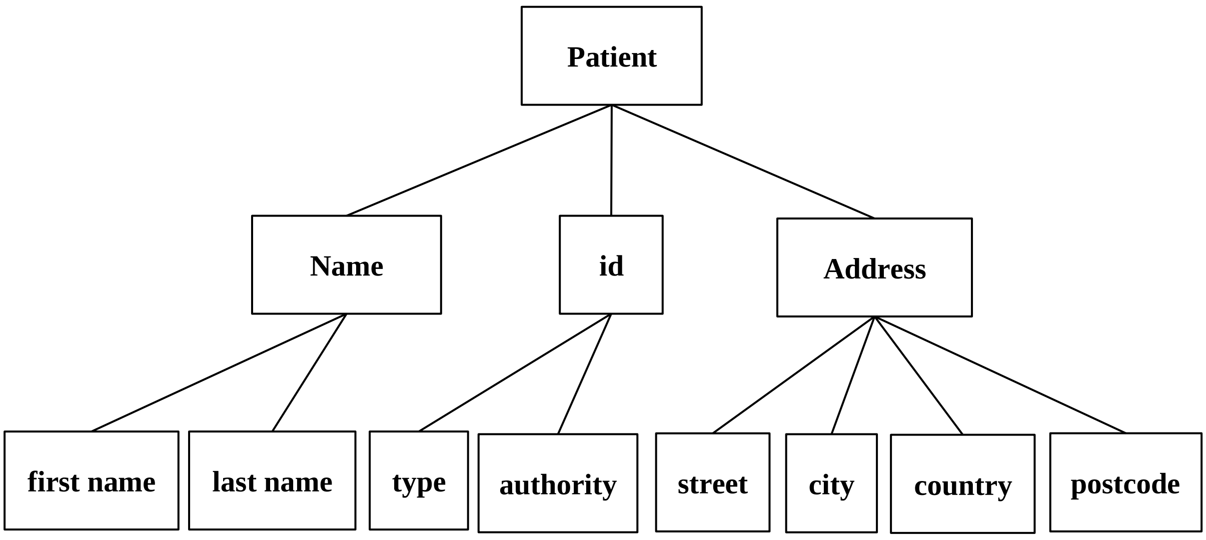


Figure 3.8 Tree for Integrated XML Schema

3.6 Performance Evaluation

The integrated schema must contain all concepts appear in any component schema completely. No redundancy in the integrated schema. The following calculations are used in the evaluations.

Recall: ‘‘It specifies the proportion of real correspondences discovered.’.

Recall = $\frac{\boldsymbol{found\_proposed\_correspondence}}{\boldsymbol{all\_proposed\_correspondence}}$ **(3.5)**

- Precision: ‘‘It reflects the proportion of genuine correspondences among all discovered correspondences.’’

Precision $= \frac{\boldsymbol{found\_proposed\_correspondence}}{\boldsymbol{all\_found\_correspondence}}$ **(3.6)**

• Precision expresses the matching's accuracy. The F measure formula is introduced to obtain a more significant statement on the quality of matchers.

F_measure = 2* $\frac{precision*recall}{preciion+recall}$ **(3.7)**

CHAPTER 4
IMPLEMENTATION OF THE PROPOSED SYSTEM

In this system, the integrated XML Schema for Heterogenous XML Schemas is produced by using SeSim method for solving semantic conflicts between two elements. StSim method is used to solve the structure conflicts between element pairs. Due to schema integration, the developers do not need to consider the semantic mapping and the structure of the organization when they have to create.

4.1 Experimental Setup

In order to implement the proposed system, install Eclipse installer 2022-23 which includes a JRE for macOS, Widows and Linux. This system evaluates the proposed methods using JAVA language. JAVA provides a rich set of libraries to create Graphical User Interface in a platform independent way. In this implementation, SWING GUI controls are used.

- **Hardware configurations**
  - Operating System: Windows 11
  - Intel(R) Core (TM) i3-6006U CPU @ 2.00GHz   1.99 GHz
  - Memory: 4 GB
- **Software requirements**
  - Eclipse version: 2022-23 version 64 bit, and
  - jre1.8.0_333 and jdk1.8.0_333

4.2 Implementation of the System

Figure 4.1 is the main page of the proposed system and there are two textboxes and nine buttons in that page. The first two buttons such as browse is for input file. The name similarity button is to find name similarity between elements in two schemas using Wordnet. The next button is presented to find element type such as simple or complex in XML schema. Cardinality occurrence is presented when the constraint button is applied. The button such as generate schema is used to solve structure conflicts between element pairs and generate a schema is used to produce final schema. The clear button is implemented to clear the previous result when next operation is started. If the system wants to close the implementation, it will use exit button.

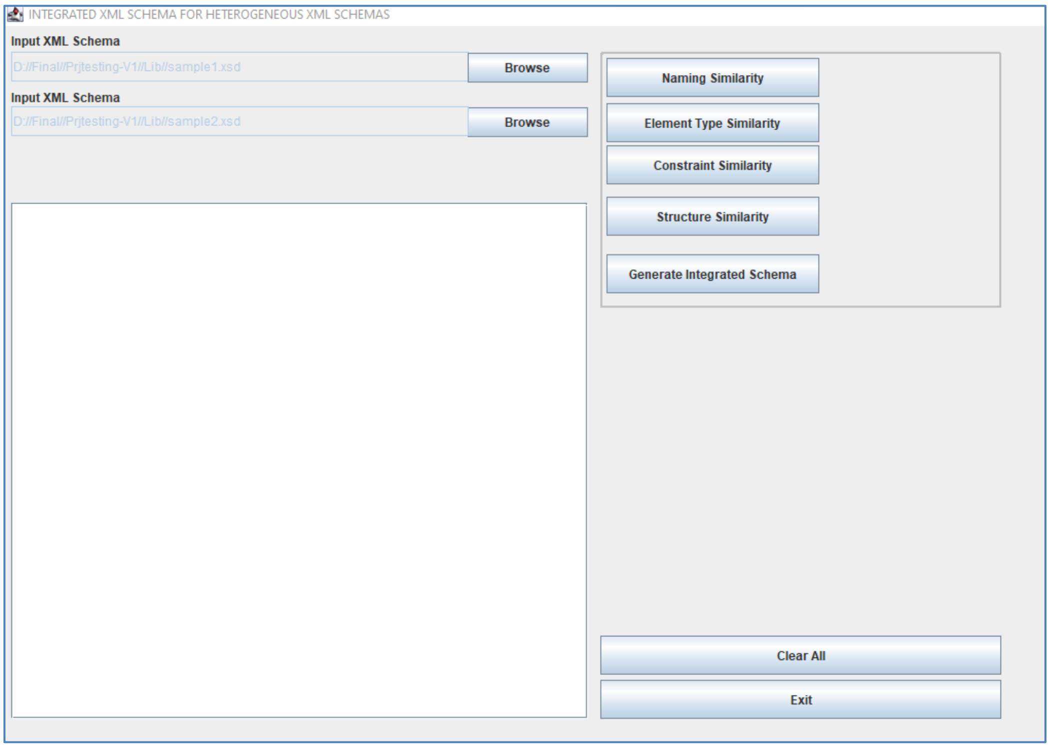

INTEGRATED XML SCHEMA FOR HETEROGENEOUS XML SCHEMAS
Input XML Schema
D://Final//Prjtesting-V1//Lib//sample1.xsd
Browse
Input XML Schema
D://Final//Prjtesting-V1//Lib//sample2.xsd
Browse
Naming Similarity
Element Type Similarity
Constraint Similarity
Structure Similarity
Generate Integrated Schema
Clear All
Exit

Figure 4.1 Main Section of the Proposed System

Firstly, the system needs to browse and upload two heterogenous XML schemas of healthcare system as input. There are two XML Schema files in the dataset as shown in the following figure.

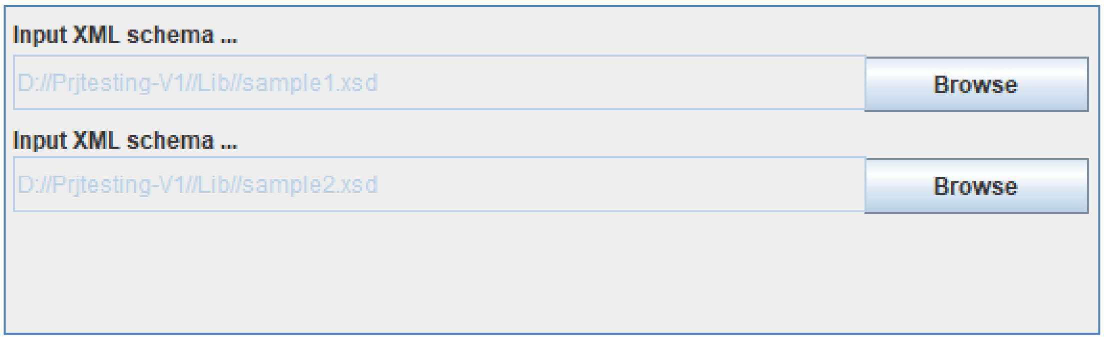
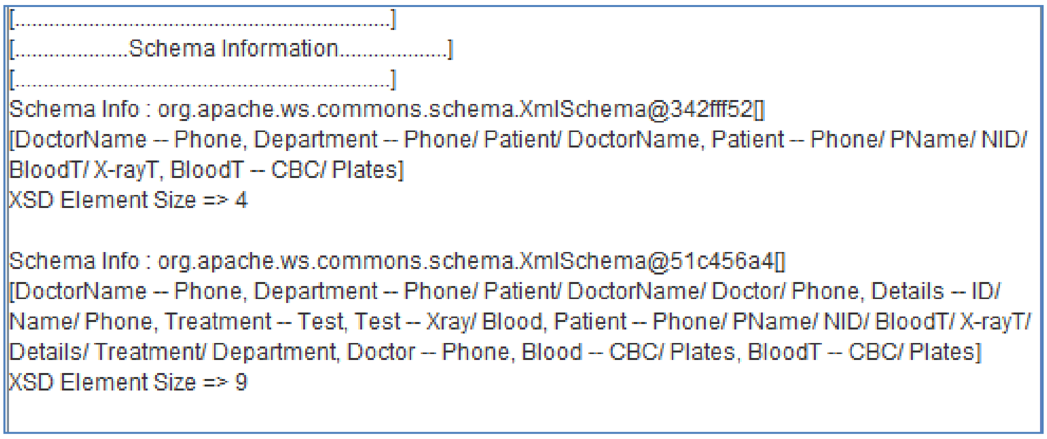

Input XML schema ...
D://Prjtesting-V1//Lib//sample1.xsd
Browse
Input XML schema ...
D://Prjtesting-V1//Lib//sample2.xsd
Browse

Figure 4.2 Two XML Schemas as Input Files

The next step is to find naming similarity each two elements between two XML schemas. When the user browses the two input files, the user can find matching element using the following button Naming Similarity. As soon as the user clicked the button, the following out will be appeared. Firstly, the system extracts subschema for each schema.

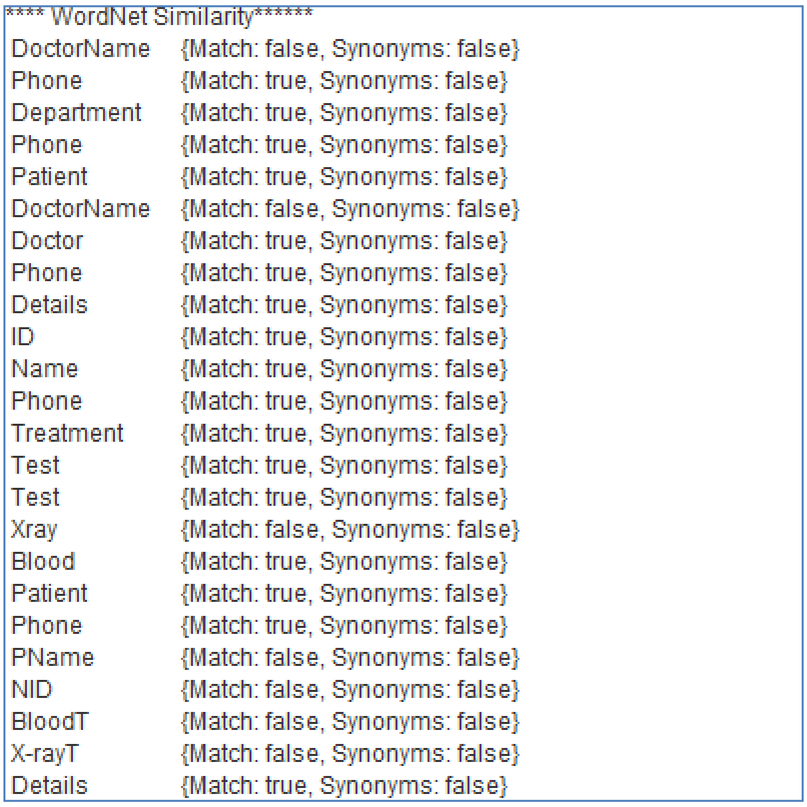


```
[...................................................................]
[....................Schema Information...................]
[...................................................................]
Schema Info : org.apache.ws.commons.schema.XmlSchema@342fff52[]
[DoctorName -- Phone, Department -- Phone/ Patient/ DoctorName, Patient -- Phone/ PName/ NID/
BloodT/ X-rayT, BloodT -- CBC/ Plates]
XSD Element Size => 4

Schema Info : org.apache.ws.commons.schema.XmlSchema@51c456a4[]
[DoctorName -- Phone, Department -- Phone/ Patient/ DoctorName/ Doctor/ Phone, Details -- ID/
Name/ Phone, Treatment -- Test, Test -- Xray/ Blood, Patient -- Phone/ PName/ NID/ BloodT/ X-rayT/
Details/ Treatment/ Department, Doctor -- Phone, Blood -- CBC/ Plates, BloodT -- CBC/ Plates]
XSD Element Size => 9
```

Figure 4.3 Generate Subschemas of Two XML Schemas

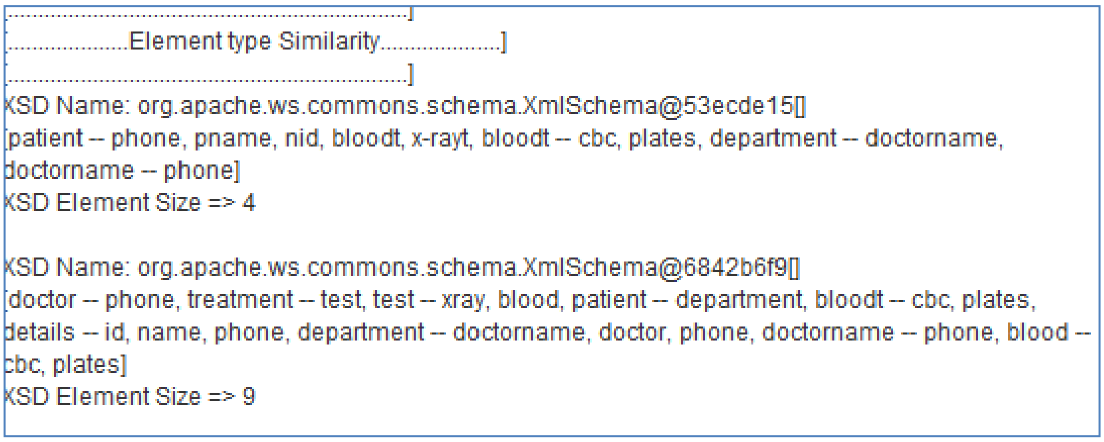
The system will check whether the first element of file one is in the file two or not using. If it is found, the similarity is 1, if not, the system put that element as new element. To handle the abbreviation of names (linguistic similarity), the system uses the WordNet to determine whether these names are in Wordnet database, or these two elements are synonyms or not that shown in Figure 4.4.



```
**** WordNet Similarity******
DoctorName    {Match: false, Synonyms: false}
Phone         {Match: true, Synonyms: false}
Department    {Match: true, Synonyms: false}
Phone         {Match: true, Synonyms: false}
Patient       {Match: true, Synonyms: false}
DoctorName    {Match: false, Synonyms: false}
Doctor        {Match: true, Synonyms: false}
Phone         {Match: true, Synonyms: false}
Details       {Match: true, Synonyms: false}
ID            {Match: true, Synonyms: false}
Name          {Match: true, Synonyms: false}
Phone         {Match: true, Synonyms: false}
Treatment     {Match: true, Synonyms: false}
Test          {Match: true, Synonyms: false}
Test          {Match: true, Synonyms: false}
Xray          {Match: false, Synonyms: false}
Blood         {Match: true, Synonyms: false}
Patient       {Match: true, Synonyms: false}
Phone         {Match: true, Synonyms: false}
PName         {Match: false, Synonyms: false}
NID           {Match: false, Synonyms: false}
BloodT        {Match: false, Synonyms: false}
X-rayT        {Match: false, Synonyms: false}
Details       {Match: true, Synonyms: false}
```

Figure 4.4 Name Similarity of XML Schemas Using Wordnet

Every element in the XML Schema document is either simple or complex in nature. If two elements have the same name and the same datatype properties. Every element in the XML Schema document is either simple or complex in nature. When two elements have the same name and their children are the same, their semantic similarity may be greater than in other cases, such as simple and complex. Because the complex element contains children, the system must compare the similarity of their children in order to compute the similarity between two complex elements. When user clicks “Element type similarity” in the main page, the system appears the following output in Figure 4.5.

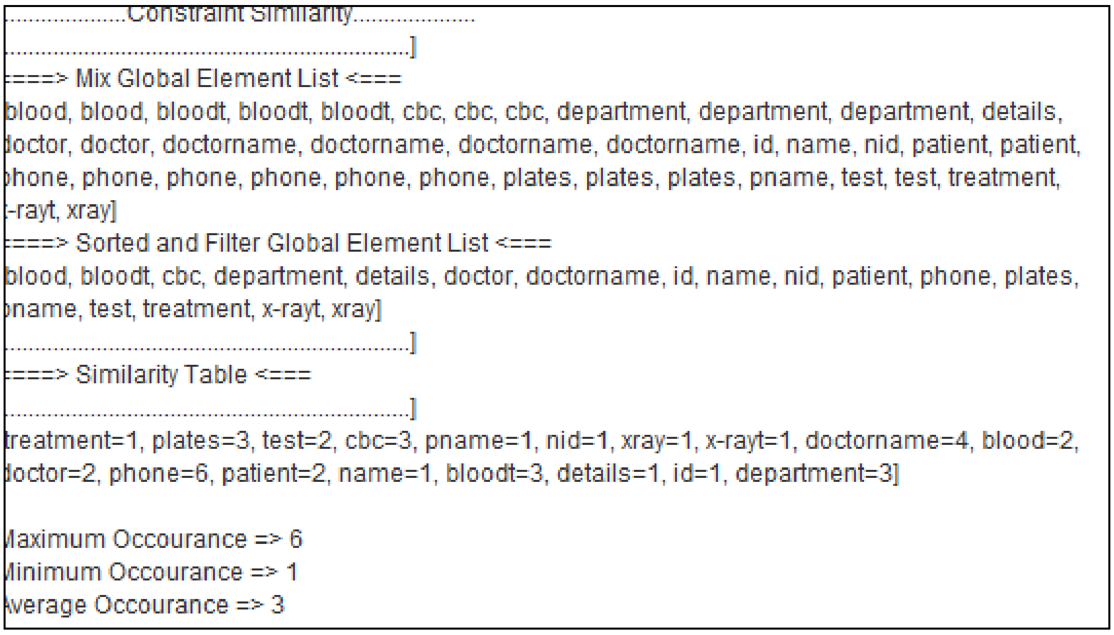


```
[...................................................................]
[...................Element type Similarity...................]
[...................................................................]
XSD Name: org.apache.ws.commons.schema.XmlSchema@53ecde15[]
[patient -- phone, pname, nid, bloodt, x-rayt, bloodt -- cbc, plates, department -- doctorname,
doctorname -- phone]
XSD Element Size => 4

XSD Name: org.apache.ws.commons.schema.XmlSchema@6842b6f9[]
[doctor -- phone, treatment -- test, test -- xray, blood, patient -- department, bloodt -- cbc, plates,
details -- id, name, phone, department -- doctorname, doctor, phone, doctorname -- phone, blood --
cbc, plates]
XSD Element Size => 9
```

Figure 4.5 Element Type Similarity of XML schemas

If the user clicks the “Constraint Similarity” link in page, the system appears with the detailed description of Constraint Similarity dataset as shown in Figure 4.6. The system computes total element list from all two schemas and then it sorts and filters element lists for. After sorting, it will calculate the occurrences for constraint similarity. The maximum, minimum and average number of occurrences will be calculated in the system.

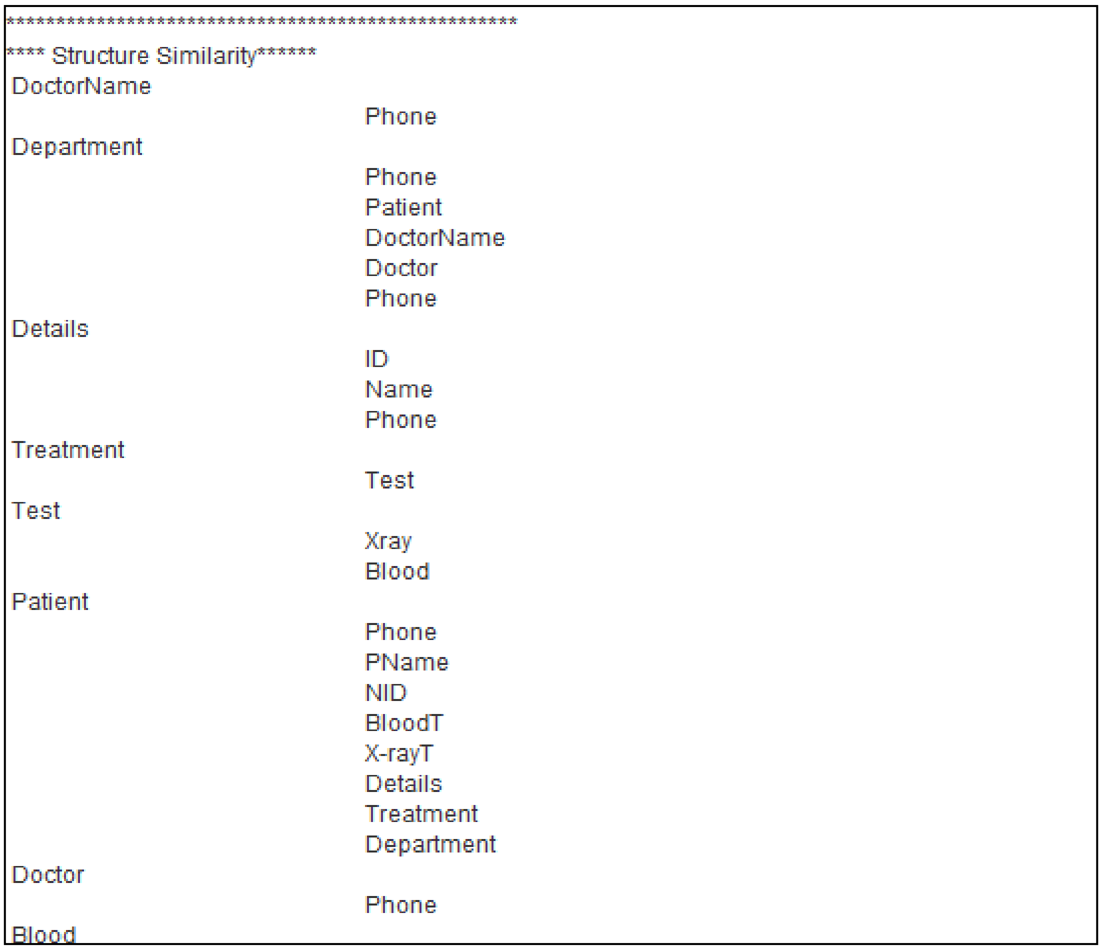


```
..................Constraint Similarity....................
...................................................................]
====> Mix Global Element List <===
blood, blood, bloodt, bloodt, bloodt, cbc, cbc, cbc, department, department, department, details,
loctor, doctor, doctorname, doctorname, doctorname, doctorname, id, name, nid, patient, patient,
phone, phone, phone, phone, phone, phone, plates, plates, plates, pname, test, test, treatment,
-rayt, xray]
====> Sorted and Filter Global Element List <===
blood, bloodt, cbc, department, details, doctor, doctorname, id, name, nid, patient, phone, plates,
pname, test, treatment, x-rayt, xray]
...................................................................]
====> Similarity Table <===
...................................................................]
treatment=1, plates=3, test=2, cbc=3, pname=1, nid=1, xray=1, x-rayt=1, doctorname=4, blood=2,
loctor=2, phone=6, patient=2, name=1, bloodt=3, details=1, id=1, department=3]

laximum Occourance => 6
linimum Occourance => 1
verage Occourance => 3
```

Figure 4.6 Constraint Similarity of XML Schemas

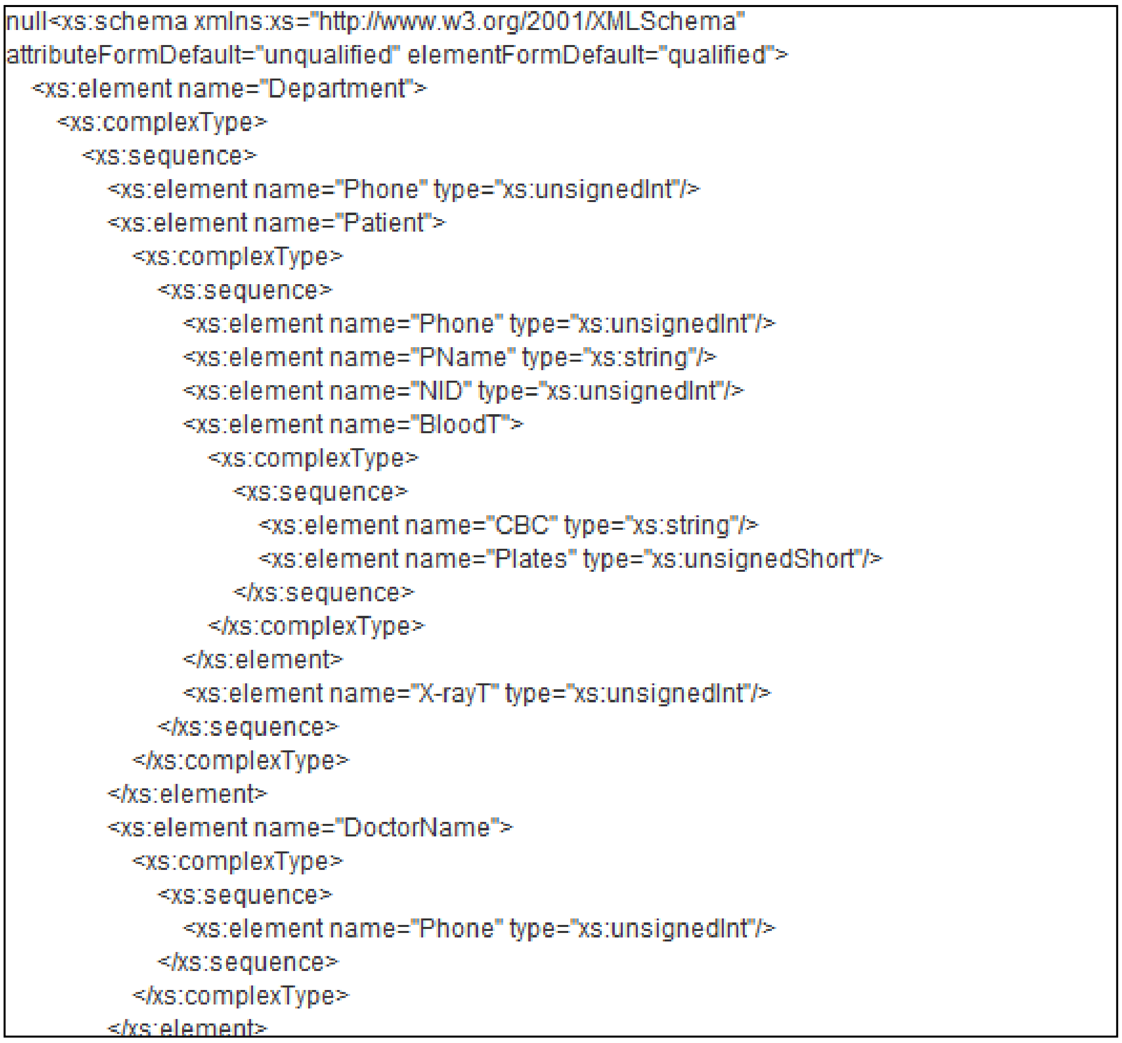
If the system would like to solve the structure conflicts of two schemas, need to click "Structure Similarity" button to display result of structure similarity output such as tree. In this step, the input is from semantic similarity phase and it displays similar structure between element pairs. If the element1 of source 1 has the same leaves as element1 of source 2, they will have same structure similarity. If not, the new elements will be added to target source.



```
****************************************************
**** Structure Similarity******
DoctorName
                                    Phone
Department
                                    Phone
                                    Patient
                                    DoctorName
                                    Doctor
                                    Phone
Details
                                    ID
                                    Name
                                    Phone
Treatment
                                    Test
Test
                                    Xray
                                    Blood
Patient
                                    Phone
                                    PName
                                    NID
                                    BloodT
                                    X-rayT
                                    Details
                                    Treatment
                                    Department
Doctor
                                    Phone
Blood
```

Figure 4.7 Structure Similarity of XML Schemas

The final output of this system is to generate an integrated XML Schema from two XML sources. The first source has twelve elements and the second has fifteen elements which has repetitive elements. There are twelve elements in the final schema in order to implement semantic and structure similarity methods. The final schema is complete and minimal for both sources.



```
null<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
attributeFormDefault="unqualified" elementFormDefault="qualified">
  <xs:element name="Department">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Phone" type="xs:unsignedInt"/>
        <xs:element name="Patient">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Phone" type="xs:unsignedInt"/>
              <xs:element name="PName" type="xs:string"/>
              <xs:element name="NID" type="xs:unsignedInt"/>
              <xs:element name="BloodT">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="CBC" type="xs:string"/>
                    <xs:element name="Plates" type="xs:unsignedShort"/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="X-rayT" type="xs:unsignedInt"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="DoctorName">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Phone" type="xs:unsignedInt"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
```

Figure 4.8 Integrated XML Schema

4.3 Experimental Result

The proposed system evaluates the performance on XML dataset. The performance of the system is evaluated according to the percentages such as Recall, Precision and F_measure.

To examine the performance of the system, it uses five datasets of XML Schemas which are from dataworld.com. Table 4.2 shows the sample datasets for measuring matching performance. The following sample XML schemas are from dataworld.com.

Table 4.1 Characteristics of the Sample Datasets

XML schemas Datasets	No. of elements	No. of leaf
Healthcare	90	49
University	72	30
Sales	85	37
Hospital	95	50
Movie	68	28

Table 4.2 shows the experimental results on dataset 1. The system uses three matching methods including proposed method to measure the performance. The sample schemas have total 90 elements and the method uses in this proposed match most of elements in sample schemas and less miss elements than other methods such as XCLust [11] and XMLSim[13]. this proposed system is higher than that of XMLSim and XCLust. The reason for this is that XClust's element similarity measurement did not consider element similarity between two elements, whereas some element pairs have the same name but differ in datatype. The XMLSim focused too much on information content similarity and ignored datatype and cardinality constraint similarity between two elements. As a result, it receives the lowest values in all calculations.

Table 4.2 Experimental Results on Dataset 1

Methods	Match Elements	Miss Elements
XCLust	70	20
XMLSim	67	23
ESim	75	15

Table 4.3 shows the experimental results on dataset 1. The system uses three matching methods including proposed method to measure the performance. The sample schemas have 72 total elements and the method uses in this proposed match most of elements in sample schemas and less miss elements than other methods such as XCLust

and XMLSim. This proposed system is higher than that of XMLSim and XCLust. The XMLSim has lowest matching elements than others.

Table 4.3 Experimental Results on Dataset 2

Methods	**Match Elements**	**Miss Elements**
XCLust	52	20
XMLSim	48	24
ESim	56	16

Table 4.3 shows the experimental results on dataset 1. The system uses three matching methods including proposed method to measure the performance. The sample schemas have 85 total elements and the method uses in this proposed has 68 matched elements in sample schemas and less miss elements than other methods such as XCLust and XMLSim. This proposed system is higher than that of XMLSim and XCLust.

Table 4.4 Experimental Results on Dataset 3

Methods	**Match Elements**	**Miss Elements**
XCLust	60	25
XMLSim	58	27
ESim	68	17

To evaluate the mediated-schema quality, the system compares the five datasets in the same manner: they all process two source schemas at a time. In general, ESim produces the integrated schema that has higher precision, recall and F-measure than XMLSim and XClust that has shown in the following chart.

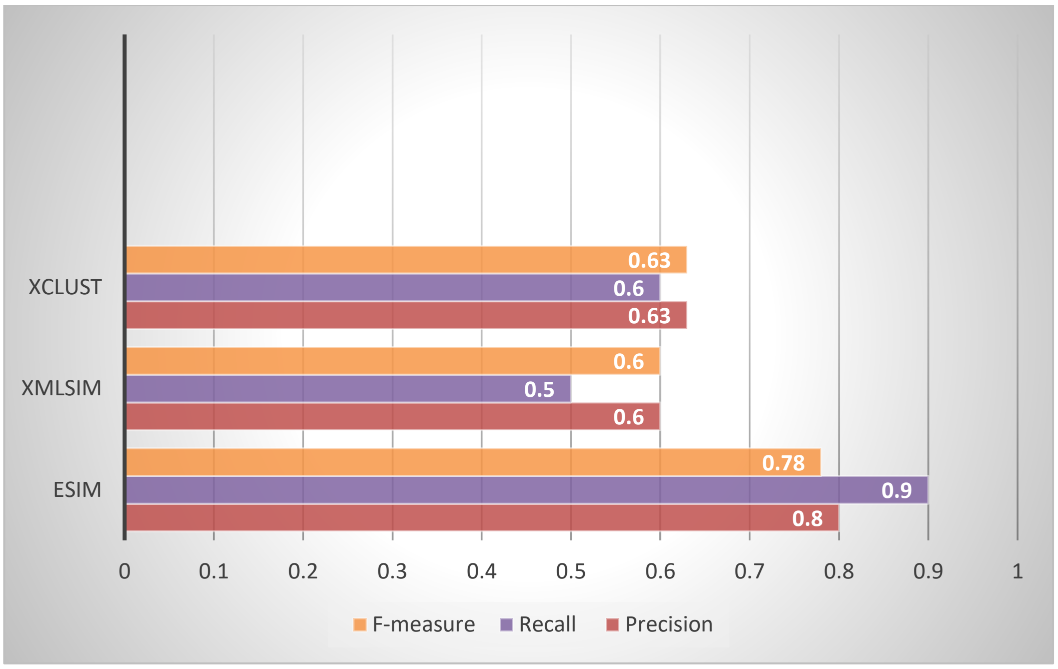


Figure 4.9 Matching Comparison of Esim with XMLSim and XClust

CHAPTER 5
CONCLUSION

XML is a major player in the exchange of data and information, playing a central role in applications. This system is generated complete and minimal integration of XML Schema among a set of heterogeneous XML Scheme sources. The system is presented a semantic and structure similarity approach for XML Schemas. It is a calculated element type similarity for XML elements and determining cardinality constraint resemblance between two elements. When two elements have semantic similarity, it is included not just language similarity but also element datatype and constraint compatibilities. The structure similarity method is used to display the distance between two elements.

The advantages of the proposed system are that it can implement in any domain area and it has three types of similarity method to find element matching. The proposed system produces an integrated schema based on structure and semantic. The disadvantages of the system is that it has missing elements and restricts hierarchical schemas.

5.1 Limitations and Further Extensions

This system can only be used in a structure heterogenous source dataset such as XSD format. The datatype constraints for attribute was not considered. The system restricts more than 1000 elements to hierarchical schemas.

In future work, this system is looking to enhance the method by addressing the semantic conflicts of overlapping concepts in source schemas and to compare the similarity of different models by measuring the similarities of unstructured data.

AUTHOR'S PUBLICATION

[1] Htun EiEi San, Thidar Win, "Integrated XML Schema for Heterogeneous XML Schemas", in the Proceedings of the (Paper ID-03014, Accepted Date: 24th June 2022) Conference on Parallel and Soft Computing (PSC 2022), Yangon, Myanmar, 2022.

REFERENCES

[1] A. Algergawy, R. Nayak, G.Saake, "Element Similarity Measures in XML Schema Matching", 2010.

[2] A.Fernandez, A. Polleres A, S.Ossowski , "Towards Fine-grained Service Matchmaking Service Matchmaking by Using Concept Similarity", Workshop on service matchmaking and resource retrieval in the semantic web, pp 31–45, 2007.

[3] C. Batini, M. Lenzerini, S.B.Navathe, "A Comparative Analysis of Methodologies for Database Schema Integration", ACM Comput. Surv. 18 (4), 323–364, 1986.

[4] D. Aumueller, H.H.Do, S. Massmann, E.Rahm, "Schema and Ontology Matching with COMA++", Department of Computer Science, University of Leipzig Augustusplatz 10/11, Leipzig 04103, Germany,2005.

[5] D.D.Yang, M.W.David, "Measuring Semantic Similarity in the Taxonomy of WordNet", The 28th Australasian computer science conference (ACSC 2005), pp 315–322, 2005.

[6] H. Ahmed, A. Hamad, "XML-Based Data Exchange in the Heterogeneous Database (XSEHD)", Department of Information Technology, College of Computer, Qassim University, 2015.

[7] H.H.Do, "Schema Matching and Mapping-based Data Integration", Dept of Computer Science, University of Leipzig, Germany, 2006.

[8] H.H.Do, E.Rahm, "COMA: A System for Flexible Combination of Schema Matching Approaches", Proceedings of the very large data bases conference (VLDB), pp 610–621, 2002.

[9] H.Q. Nguyena, D.Taniar, J. W. Rahayua, K. Nguyena, "Double-layered schema integration of heterogeneous XML sources", a Dept. of Computer Science and Computer Engineering, La Trobe University, VIC 3086, Australia, 2011.

[10] I. Brown, A. Adams, "The Ethical Challenges of Ubiquitous Healthcare", Int Rev Inf Ethics 8(12):53–60, 2007.

[11] M.L.Lee, L.H. Yang, W. Hsu, X. Yang, "XCLust: Clustering XML Schemas for Effective Integration", ACM Press, New York, pp 292–299, 2002.

[12] M. M. Thu, "Clustering XML Documents using XEdge Algorithm", Conference on Parallel and Soft Computing, Yangon, Myanmar, 2015.

[13] P. Resnik, "Semantic Similarity in a Taxonomy an Information-Based Measure and its Applications to Problems of Ambiguity in Natural Language", J Artif Intell Res 11:95–130,1999.

[14] S. Bechhofer, R. Volz, and P. Lord "Cooking the Semantic Web with the OWL API", International Semantic Web Conference (ISWC), 2003.

[15] X.Yang, M. Li Lee, and T. Wang Ling, "Resolving Structural Conflicts in the Integration of XML Schemas: A Semantic Approach", School of Computing, National University of Singapore Science Drive 2, 2003.

[16] X. Xu, T. Wang, Y. Yang, L. Zuo, F. Shen, H.T. Shen, "Cross-Modal Attention With Semantic Consistence for Image-Text Matching", IEEE Transactions on Neural Networks and Learning Systems, 2020.

[17] Y. Cheng, Z.Wu, "Design of Digital Campus Platform Data Integration System Based on XML", International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), 2019.

[18] https://www.comp.nus.edu.sg/~lingtw/cs4221/integration.pdf

[19] http://en.wikipedia.org/ wiki/Electronic_health_record

[20] Mebiquitous XML Schema. http://ns.medbiq.org/

[22] http://www.oagi.org/ dnn2/DownloadsandResources.aspx.