XML Gateway for Heterogeneous Distributed Database Management Systems

Aslam Parvez Memon and Muhammad Nadeem
SZABIST
Karachi, Pakistan

Abstract:
Conventional standalone application development paradigm is being transformed into multi-layered application development process. The main reason of transformation is the execution environment— particularly internetworking. The above mentioned transformation requires major changes in the platforms i.e. Hardware, Software, application development tools, and database management systems.

This study especially focuses on the data related issues of applications, running on distributed platforms. The key attributes of distributed applications' data include Autonomy, Distribution and Heterogeneity.

Autonomy of DBMS can be defined as a degree to which a database management system can operate independently, which can be implemented at design, communication and execution level. Distribution deals with data distribution on diverse network platforms.

Database Management System components can vary in Data Model, Data Structure, Query Languages, Data Manipulation, Schema, Operating System, Hardware, Location. The Databases that differ in these are termed as Heterogeneous Databases.

Today's applications often need integration of databases deployed on variety of platforms and possibly supporting different data models and data structures. It seems very difficult to implement the above stated attributes in different database management systems provided by various vendors. In order to integrate the information from heterogeneous data sources, it is necessary to form a gateway or mediator architecture.

*XML (e**X**tensible **M**arkup **L**anguage) is a portable data definition and transportation language. One of the major applications of XML is to provide the mediation between two or more heterogeneous databases.*

This study aims at designing a prototype for accessing heterogeneous data sources using XML standard. Additionally the comparative study of Databases and XML has been carried out to present XML as potential candidate for data modeling and data definition requirements of any application.

Furthermore an interactive data cleaning, analysis and transformation model has been added to integrate the diverse data and perform variety of data cleaning operations.

1. INTRODUCTION

In today's distributed computational environment, Business to Business or Business -Consumer data requirements can not be satisfied by enterprises' centralize data repositories. Enterprises may have to opt for the data distributed across distant geographical locations, or from other commercial data stores. The data obtained from distributed geographical locations may not compatible, and may need fine-tuning in order to make it suitable as per applications' requirements. A Middle-level organization may not afford the expensive Heterogeneous Database Management System tools, in order to reformat, regenerate or represent the data for business requirements. The proposed XML based gateway architecture would address the data centric problems of enterprises by providing the light weight application tool for data transmission, transformation, cleaning and filtration at one place. Additionally the use of XML as a medium provides the platform independent, portable solution. An Interactive data cleaning, analysis and transformation model also has been added to integrate the diverse data and perform variety of data cleaning operations

1.1 What is XML?

XML (e**X**tensible **M**arkup **L**anguage) is a portable data definition and transportation language. XML specifications were released in February, 1998 by ***W3C***, its license free, platform independent and well supported standard. XML facilitates Metadata (Data about Data) definitions, and focuses on creating new tags and elements according to the need of data centric applications. XML files contain the information about definition, formatting, structuring, relation (with other data files) and distribution of content--- its all *data*, *data* and *data*.

XML is emerging as standard data transportation protocol, to be used by diverse applications running on distributed heterogeneous platforms, providing data independence. XML is lingua franca for data transportation. XML is portable in a sense that XML specifies Unicode as its default character set. Unicode provides room for over one million different characters.

Virtual Document Server for Digital Libraries [1] extracts Bibliographical data and documents as XML documents and stores as files at the operating system level. The only function that can be used is the retrieval of documents via a search engine or other information retrieval tools.

XML describes structure and semantics not formatting [2]. XML documents contain user-defined tags, along with their syntactic and semantic information. XML file containing Students' Information might look like this:

```
<? Xml version="1.0" encoding="ISO8859-1"?>
<StudentInfo>
<Student>
        <Student Name="Jameel">
        <RegNo>1</RegNo>
        <Phone>678555</Phone>
        <Subject>Computer Science</Subject>
</Student>
<Student>

        <Student Name="Rehan">
        <RegNo>2</RegNo>
        <Phone>568555</Phone>
        <Subject>Management Science</Subject>
</Student>
<Student>
        <Student Name="Raheel">
        <RegNo>3</RegNo>
        <Phone>585555</Phone>
        <Subject>Social Science</Subject>
 </Student>
</StudentInfo>
```

1.2 Web Applications of XML

The applications that will drive the acceptance of XML are those that cannot be accomplished within the limitations of HTML [3]. These applications can be divided into four broad categories:

1. Applications that require the Web client to mediate between two or more heterogeneous databases.

2. Applications that attempt to distribute a significant proportion of the processing load from the Web server to the Web client

3. Applications that require the Web client to present different views of the same data to different users.

4. Applications in which intelligent Web agents attempts to tailor information discovery to the needs of individual users.

The alternative to XML for these applications is proprietary code embedded as "script elements" in HTML documents, and delivered in conjunction with proprietary browser plug-ins or Java applets. XML derives from a philosophy that data belongs to its creators and that content providers are best served by a data format, that does not bind them to particular script languages, authoring tools, and delivery engines, but provides a standardized, vendor-independent, level playing field upon which different authoring and delivery tools may freely compete.

An example of the first category of XML applications could be an information tracking system for a home health care agency. This application could then have the following functions that are not all accomplishable in HTML:

1. Log into the hospitals web site.

2. Access the patient's medical records in a Web-based interface that represents the records for that patient with a folder icon.

3. Drag the folder from the application over to the internal database.

4. Drop it into the database.

The application could use XML tags such as <allergies>, <drug-reaction>, and so on.

1.3 XML and Data Portability

Personal Computers, Mainframes, Mini-frames, Macs and other computer systems interconnected with internet use different character sets, the data transportation process that is the major application of XML has to take care of various character sets specially for trading files/documents/ data on internet with anyone.

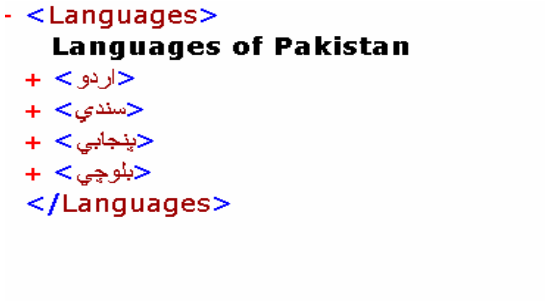
XML specifies Unicode as its default character set. Unicode provides room for over one million different characters. Currently a few more than 94,000 different Unicode characters are defined.

Unicode provides a unique number for every character, no matter what the platform, no matter what the program, no matter what the language [4].

These encoding systems also conflict with one another. That is, two encoding can use the same number for two *different* characters, or use different numbers for the *same* character. Any given computer (especially servers) needs to support many different encoding; yet whenever data is passed between different encoding and platforms, that data always runs the risk of corruption.

The Unicode worldwide character standard is a character coding system designed to support the interchange, processing, and display of the written texts of the diverse languages of the modern world. In addition, it supports classical and historical texts of many written languages

The inherent support of Unicode in XML enables developers to create international XML documents. The author has implemented Unicode specifications for writing XML files containing formatted data of Pakistani Local languages, a typical well formed XML containing the sentences of Urdu, Punjabi, Sindhi, Pushto and Baluchi is given below:



The e**X**tensible **M**arkup **L**anguage is emerging as a universal mechanism for data sharing, which provides common format for expressing the data structure and content. XML has removed major obstacles to sharing or migrating data among diverse applications and databases due to its robust platform independent characteristics.

XML with XSL (*eXtensible Stylesheet Language; a way of defining style sheets in XML*) can be used to address the needs of content and formatting styles across applications and platforms.

1.4 Transforming Unicode XML data to HTML

An XSL stylesheet consists of a set of rules that determine how specific elements from the XML document should be processed [5]. Each rule has a match criterion that specifies the pattern of elements for which the rule is activated. Each rule also has an associated transformation that acts on the matching elements. To transform the following Unicode XML document into HTML

```
<?xml version="1.0" ?>
<?xml-stylesheet type="text/xsl" href="contacts.xsl"?>
<CONTACT_INFO> Information Of Contacts
<CONTACT>
<Name>اسلم پرويز ميمن </Name>
<Address>حيدرآباد</Address>
<Class> ايم ايس</Class>
</CONTACT>
<CONTACT>
<Name>محمد نديم</Name>
<Address>كراچي</Address>
<Class>ايم ايس  </Class>
</CONTACT>
<CONTACT>
<Name> سجاد رضا عابدي</Name>
<Address>كراچي</Address>
<Class>ايم ايس  </Class>
</CONTACT>
<CONTACT>
<Name>عاصم</Name>
<Address>كراچي</Address>
<Class>ايم ايس  </Class>
</CONTACT>
<CONTACT>
<Name> علي صمد </Name>
<Address>كراچي</Address>
<Class>ايم ايس  </Class>
</CONTACT>
</CONTACT_INFO>
```

Consider the following XSL document as an HTML template to populate a HTML document with XML data.

```
<?xml version= "1.0"?>
<xsl:template xmlns:xsl="http://www.w3.org/TR/WD-xsl">

<html>
<Body>
<h1 align="Center">
<Font color="Red"> فهرست  تعلقات </Font>
</h1>
<table align="center" border="1">
<xsl:for-each select = "CONTACT_INFO/CONTACT">
<tr>
<td>
<b><xsl:value-of select="Name"/></b>
</td>
<td>
<b><xsl:value-of select="Class"/></b>
</td>
<td>
<b><xsl:value-of select="Address"/></b>
</td>
</tr>
</xsl:for-each>
</table>
</Body>
</html>
</xsl:template>
```

Which yields the out put as follows:

Figure 1: The HTML version of the Unicode XML document.
Source*: Software Prototype Designed by Author*

2. XML GATEWAY FOR HETEROGENEOUS DISTRIBUTED DATABASE MANAGEMENT SYSTEMS

In today's distributed computational environment, Business to Business or Business -Consumer data requirements can not be satisfied from enterprises' centralize data repositories. Enterprises may have to opt for the data distributed across distant geographical locations, or from other commercial data stores. The data obtained from distributed geographical locations may not compatible, and may need fine-tuning in order to make it suitable as per applications' requirements. The following type of semantic heterogeneity characterize schema of independently developed data sources [6]:

- Name *heterogeneity* regarding name conflicts in different sources schemas (synonyms and homonyms)
- Structural *heterogeneity* concerning the representation of the information in different schemas. Structural heterogeneity mainly include: 1) *Type conflicts* representing the same piece of information using different constructs in different schemas. 2) *Key conflicts* assigning different keys to the same concepts in the different schemas 3) *domain conflicts* regarding compatibility in domain values.

A Middle-level organization may not afford the expensive Heterogeneous Database Management System tools in order to reformat, regenerate or represent the data for business requirements.

The extensible markup language XML is data storage language. It is a particularly simple data storage language often used to temporarily store data while it is transferred from one application to another. It stores data in a format defined by a series of straightforward tags created by programmer.
By itself XML is pretty feeble because it is designed only to store data and provides no features for manipulating it. In contrast, Access, Oracle, Informix, SQL Server, and a host of other database systems provide powerful data manipulation features such as indexing, searching and sorting. XML has only two advantages over these powerful database systems. It is simple, and it is portable. These may seem like minor benefits, but together they allow XML to add a powerful new tool to data handling enterprises.

XML syntax is easy to learn and use. It allows a programmer to quickly define complex hierarchical data structure that may contain other hierarchical data structures. Even though it is possible to build this kind of data structure using other database products, their rigid table structures force the programmer to use complicated techniques to model these data relationships. Because XML stores data definitions and values in ordinary text files, programmer can create and modify them quickly and easily.

Less glamorous but probably more important is XML's portability. XML is easy to use in any environment that includes an XML parser. Visual Basic .NET includes extensive tools for manipulating XML files. These tools allow Visual Basic .Net programs to read XML files created by programs using another parser. Conversely, these tools allow a Visual Basic .Net program to create XML files for use by other programs that will use another parser to read them.
By running on a variety of different platforms, XML positions itself to become the lingua franca of data. Most large companies have distinct organizations that hold different key pieces of business data. Separate parts of a telephone company, for example, hold information about customers, billing, central office hardware, switch software, telephone numbers, outside plants (cables, poles, repeaters), and so forth. The data is typically stored on an assortment of hardware and software platforms scattered around the company. Trying to gather data from these diverse sources for reporting and analysis is a monumental task.

XML changes that. Each part of the business can build tools to save data in XML format. Once all of the data has been saved in the common format, reporting and analysis becomes trivial. Furthermore, a programmer who has a platform with an XML parser can easily read and manipulate the XML data saved by other parts of the company. Programs can reach across company networks or even the internet to gather this information from wherever it is stored and produce a unified report. XML provides a solid framework that let programmers store data in a format that any other application can understand.

The World Wide Web also provides a new opportunity for sending data from many data sources to a variety of applications and output devices. Because so many companies want the ability to display data on the web, tools have been developed to display XML. Using a simple transformation file is XSLT and

XSLTransformation class; a Visual Basic .NET application can transform data into HTML for display on internet or on intranet. Voice XML for use in speech applications or any the text based format.

XML is emerging as standard data transportation protocol, being used for providing data independent and fulfilling the data centric requirements of applications running on distributed heterogeneous platforms, using heterogeneous database management systems. XML is lingua franca for data transportation.

Moving from one application to another is an extremely common programming task. Users doing their jobs with a variety of applications generate a huge amount of data. Another application extracts that data and produces reports. The reports are pulled into yet another system to create summaries, executive briefings, and so forth. All too often this exercise means wading through a labyrinth of different file formats, database interfaces, remote procedures calls, and intermediary data formats.

The extensible markup language XML is changing all that. XML is a simple but powerful language that lets you store data in XML files effortlessly. By standardizing data storage, XML lets any number of applications share data with little effort on the programmer's part. Data gathered by one application can be shared with other applications running anywhere on the Internet or on a private network. Tools for working with XML fields have been around for a while now, but the new release of Visual Basic .Net raises this support to a new level. Previously tools were tacked on to the Visual Basic. In Visual Basic .Net, XML support is tightly integrated into the language and provides many new methods for manipulating XML data. This tight integration means Visual Basic programmers can load, manipulate, and save XML data faster and more easily than ever before.

The proposed architecture of XML gateway for Heterogeneous Database Management Systems Figure: 2 serve as a middleware between various heterogeneous distributed database management systems and XML repositories.

2.1 The Architecture

Users want seamless access to all relevant information about their domains' real world objects distributed across platforms and applications [7]. Various approaches have been adopted to achieve the goal, which includes Integration within the applications, Data warehouses, Federated Databases, Messaging, and Parameter Passing, schema integration, multi-databases, systems and mediator based systems [6].

The proposed mediation Architecture is a light weight XML based solution, comprises on two major components— Component for acquiring data from heterogeneous data sources and another for data cleaning, analysis and transformation. This section deals with data acquisition part of the architecture.

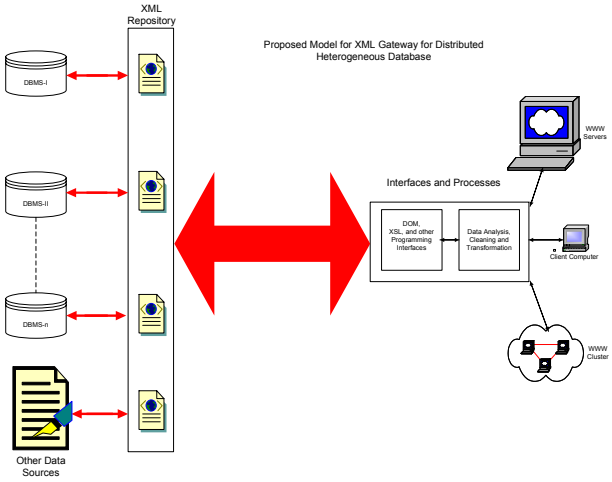


Figure 2: XML Gateway Architecture of Heterogeneous Management Database Systems
Source*: Software Prototype Designed by Author*

The data acquisition part enables the users to acquire data from heterogeneous sources. The heterogeneous sources include: ODBC sources, Text Documents, Email messages, and spreadsheet data etc. Additionally the Gateway provides seamless connectivity mechanism to heterogeneous data sources generating the merged semi-structured XML data files. XML is universally accepted data formatting and transportation languages. The XML data files generated by the Gateway can also be used by diverse platforms and applications across the networks.

An XML document is a database only in the strictest sense of the term. That is, it is a collection of data. In many ways, this makes it no different from any other file -- after all, all files contain data of some sort. As a "database" format, XML has some advantages. For example, it is self-describing (the markup describes the data), it is portable (Unicode), and it can describe data in tree or graph structures. It also has some disadvantages. For example, it is verbose and access to the data is slow due to parsing and text conversion [8].

A more useful question to ask is whether XML and its surrounding technologies constitute a "database" in the looser sense of the term -- that is, a database management system (DBMS). The answer to this question is, "Sort of." On the plus side, XML provides many of the things found in databases: storage (XML documents), schemas (DTDs, XML schema languages), query languages (XQuery, XPath, XQL, XML-QL, QUILT, etc.), programming interfaces (SAX, DOM, JDOM), and so on. On the minus side, it lacks many of the things found in real databases: efficient storage, indexes, security, transactions and data integrity, multi-user access, triggers, queries across multiple documents, and so on [9]. *XML and other*

structures integration [10] further describe the role of integrating XML with other data sources.

Keeping in view the potential use of XML as counterpart of platform independent, universal, portable database having all characteristics of a typical database management system the XML based Gateway for Heterogeneous database management system has been proposed in the study.

Heterogeneous data exposes variety of challenges, in order to provide the uniform data view to users'/ consumers' certain reconciliation and cleaning operations are required to be performed. The proposed architecture performs dual operations of data transformation from heterogeneous sources to XML and Data reconciliation.

XML can play a substantial role in providing platform independence in data sharing process. Major reasons of heterogeneity in data include: variation/ distribution of Data Model, Data Structure, Query Languages, Data Manipulation, Schema, Operating System, Hardware and Location. XML has the potential to address the vulnerabilities caused by heterogeneity.

XML provides a neutral syntax fro describing graph-structured data as nested tagged elements with links. Because the proposed middleware architecture helps in transforming diverse data structures into such graph, which can later be reformatted as per user requirements in the data cleaning component of the architecture. The transformation of heterogeneous data sources into XML-based neutral format enables to deal with the problems of heterogeneity in data structures.

XML format is universally accepted; almost every software and hardware platform is either supporting or planning to support the XML specifications. The XML support ranges from Operating Systems, Application development software to internet browsers.

Looking at the universal acceptance of the XML in the field of data transportation, sharing and portability, researchers have contributed a lot in the formation of query languages for XML data sources; this makes the proposed architecture more viable. XQL, XML-QL, YATL, and Lorel were developed by the database community for processing the XML data. Technically the XML query languages listed above have several features that we believe are especially important [11].

As stated earlier the data may be obtained form various geographically distributed locations. Numerous standards, protocols, products and middleware technologies are available to handle the challenges exposed by geographically distributed data. W3C's SOAP: *Simple Object Access Protocol* and OMG's CORBA: *Common Object Request Broker Architecture* and HTTP: Hypertext Transfer Markup Protocol to name a few.

SOAP [Simple Object Access Protocol] and Web both make extensive use of XML to address the geographic distribution of data. SOAP is an emerging distributed middleware technology that uses a lightweight and simple XML-based protocol to allow applications to exchange structured and typed information across the web. SOAP applications can be written in a variety of programming languages, used in combination with a variety of internet protocols and formats such as HTTP, SMTP and MIME and can support many types of applications ranging from messaging systems to RPC [remote procedure calls]. Thus the data sources generated through the proposed middleware architecture can be potentially used for transportation using SOAP technology.

3. DATA CLEANING

The research work is aimed at providing a complete underlying solution for integrating heterogeneous data sources. A Business – Business or Business – Customer application needs to provide access to the enterprise data collected from various diverse sources, without letting the users to look into the syntactic and structural details. The data obtained form heterogeneous sources may have inconsistencies due to difference in data formats within distinct sources, data entry errors and no adherence to integrity constraints.

The data with these discrepancies is referred to as dirty data , the examples of dirty data include the data items with spelling errors, phonetic and typing errors, word transpositions, extra words, missing data elements, inconsistent values, multiple values in a single free-form field, misfielded values, illegal values, abbreviations, truncation and initials, etc. Consequences of dirty data may be existence of duplicate data that needs to be consolidated. The data must be reconciled and transformed into a uniform format before it can be used for the emerging data sharing requirements.

The process of accessing or sharing heterogeneous data sources exposes an array of challenges to be met. Data sources may have different levels of heterogeneity-- heterogeneous Data Model, Data Structure, Schema, attribute representation & semantics, Distributed Geographical Locations, Query Language, Data Manipulation, Operating System, and Hardware. Reconciliation or cleaning of data must address aforesaid levels of heterogeneity optimally.

3.1 Approaches to Data Cleaning

The data cleaning has three major components: Auditing data to find discrepancies, choosing transformation to fix these, and applying the transformations on the datasets [12][13].
The Data cleaning process can be carried out into two stages-- Content Level Data Cleaning and Schema Level Data cleaning:

Content level data cleaning deals with the fixing the discrepancies in the content of the data sources, the major discrepancies include the data items with:
Spelling errors, Phonetic & typing errors , Word transpositions, Extra words, Missing data elements, Inconsistent values, Multiple values in a single free-form field, Content Checking against List of Values, Content Formatting: Numeric, Alpha Numeric, Date & Time etc.
Misfielded values; and Illegal values, abbreviations, truncation and initials

Schema level data cleaning deals with fixing the discrepancies in the Schema Definition of the databases, the major discrepancies and cleaning operations include:
Inconsistency checking in schema definitions
Schema Level cleaning operations: Addition, Merging, Deleting, Splitting, Folding, Copying and Transforming Columns.
Checks for adherence with Integrity Constraints.
Content verification against schema.

3.2 The Method

The approach adopted in *Potters Wheel: an Interactive Data Cleaning System* [12][13] has been implemented for Data cleaning, Transformation and Discrepancy detection activities in the study. Potter's Wheel accepts input data from an ODBC source or any ASCII file descriptor. Keeping in view the scope of the study, potter's wheel architecture has been implemented for XML data sources. The XML files generated from heterogeneous data sources through *XML Gateway for Heterogeneous Distributed Databases* may used as an input to the cleaning, transformation and discrepancy detection system. The usage of XML as input data source has been recommended by the authors of the research paper [12][13] as future direction of work.

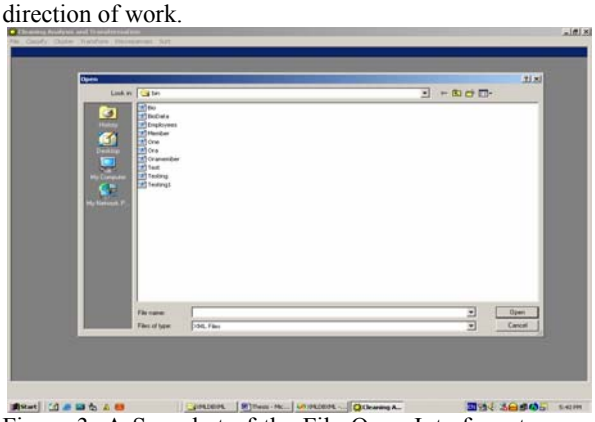


Figure 3: A Snapshot of the File Open Interface, to open XML data sources for transformations.
Source*: Software Prototype Designed by Author*

XML data is populated into spreadsheet like interface to perform the Data Cleaning, Transformation and discrepancy detection operations.

Data cleaning is no doubt a tedious task; efforts have been put to make the process interactive and user-friendly. The data populated into spreadsheet like interface as at Figure- 4, is presented to user to perform the cleaning, transformation and discrepancies detection process. Discrepancy detection algorithms have been applied; the discrepancies detected through the process are displayed using simple interfaces. User interaction is the main feature of the system; changes carried out on the records populated in the spreadsheet, and can be undone any time.

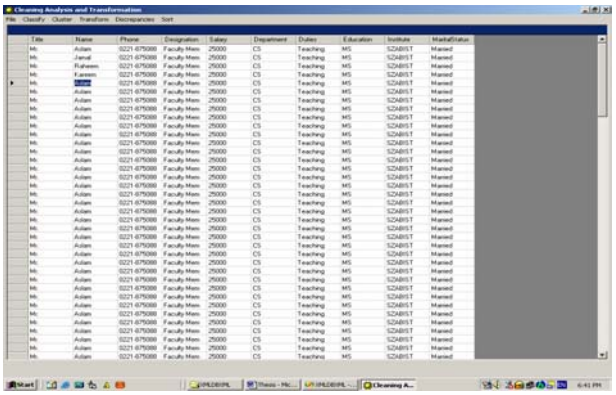


Figure 4: A Snapshot of the Spreadsheet Interface populated with XML data sources for transformations and discrepancy detection process.
Source*: Software Prototype Designed by Author*

The tasks of content level and schema level cleaning, transformation and discrepancy detection have been integrated into single interface. The process is carried out on the background data seamlessly; users do not require being involved into programming jargons.

Cleaning and transformation operations are carried out interactively on the data displayed in the spreadsheet interface, which can be committed to make the changes permanent.

3.3 Data Source

XML (e**X**tensible **M**arkup **L**anguage) is a portable data definition and transportation language, its platform independent and well supported standard. XML facilitates Metadata (Data about Data) definitions, and focuses on creating new tags and elements according to the need of data centric applications. XML files contain the information about definition, formatting, structuring, relation (with other data files) and distribution of content--- its all *data*, *data* and *data*.

XML is emerging as standard data transportation protocol, to be used by diverse applications running on distributed heterogeneous platforms, providing data independence. XML is lingua franca for data transportation. The XML Gateway for Heterogeneous Distributed Databases generates the corresponding XML representation of the database files, which serves as an input to the cleaning

system. Keeping in view the promising benefits of XML towards data independence, portability, distribution the XML has been selected as a major data source for the cleaning system. Additionally the cleaning system uses XML Schema Definition Languages as tool for describing the customized domains, making the system purely XML oriented. XSD being the native Schema definition languages for XML has obvious benefits over DTD and other Schema definition languages.

4. CONCLUSION AND FUTURE DIRECTIONS

Efforts have been put forward to come up with a light weight intermediary architecture for heterogeneous databases. The implementation of the proposed architecture would serve as a tool for wide range of services which include, data acquiring, data transportation, inter-database conversion and data cleaning. This article covers the issues of Application Integration, Conceptual Schema, Data Extraction, Data Portability, Data Representation/ Data Structure, Data Independence, External Schema, Data Integrity, Query Languages.

One important aspect of XML and related technologies that has not been addressed in this research work is the XML Security. XML Security is important for the XML Data files or intermediary data obtained from diverse platforms, keeping the security constraints in mind the XML has been proposed as an intermediary medium of storage during transformation process.

REFERENCES

[1] Andreas Heuer Holger Meyer Beate Porst, _ Patrick Titzler_ "Virtual Document Servers for Digital Libraries" Database Research Group Computer Science Department, University of Rostock 18051 Rostock, Germany, 2000

[2] Elliotte Rusty, "Harold. XML Bible, Second Edition", Hungry Minds, Inc. New York, 2001.

[3] "XML What is in it for us" http://tech.irt.org/articles/js072/1

[4] Unicode Organization's Official Website http://www.unicode.org

[5] Cristian Georgescu, Code Generation Templates Using XML and XSL UML C/C++ Users Journal http://www.cuj.com/articles/ 2002/ 0201/ 0201a/ 0201af2.htm?topic=articles

[6] Silvana Castano, Valeria De. Antonellis and Sabrina De Capitani di Vimercati "Global Viewing of Heterogeneous Data sources" *IEEE Transactions on Knowledge and Data Engineering* Volume 13 No. 2 March/ April 2001.

[7] Thomas Lohman, Bill Murray and Vijay Kanabar. "A Heterogeneous Distributed Database Architecture for University Integrated Circuit Manufacturing Research" Massachusetts Institute of Technology 50 Vassar Street Bld. 36-297 Cambridge, MA. 0`2139, Stanford University Stanford, CA. 94305, Boston University 755 Commonwealth Avenue Boston, MA. 02215

[8] Len Seligman, Arnon and Rosenthal "XML's Impact on Databases and Data Sharing" IEEE Research Feature, June 2001.

[9] Ronald Bourret "XML and Databases" June 2001.

[10] Peter McBrien, Alexandra Poulovassilis "XML and other structures... Ingeneration" Department of Computing Imperial College London, Department of Computer Science, BirkBeck college University of London.

[11] "XmL and Queury Langauges" http://www-db . research . bell-labs.com/ user / simeon /xquery.html

[12] Potter's Wheel A-B-C: An Interactive Tool for Data Analysis, Cleansing, and Transformation http://control.cs.berkeley.edu/abc/

[13] Vijayshankar Raman and Joseph M.Hellerstein "Potter's Wheel: An Interactive Data Cleaning System" University of California at Berkeley, 2001