Web Data Extraction for Developing a Mashup

Poonam. A. Chaudhari Rahul. L. Paikrao

***Abstract*— Web is a huge reservoir of information. Data available is extremely diversified and abundant. Various types of data can be easily extracted from the Internet, although not all of the data is relevant to the users. Most web pages are in unstructured HTML format, making web data extraction process very time consuming and costly. There is a necessity to convert unstructured HTML format into a new structured format such as XML or XHTML. We propose an approach for implementing web data extraction and developing a Mashup from HTML web pages. It also helps to collaborate and integrate various stages of building a Mashup, i.e., Data Retrieval, Data Source Modeling, Data Cleaning/Filtering, Data Integration and Data Visualization. The data modeling stage renders Document Object Model (DOM) tree with the help of HTML Parser. Some algorithms and rules are used so that it can specifically analyze the HTML tags and extract the data into a new format. The core algorithm can extracts web data tables using recursive technique while rendering the DOM tree model automatically. Furthermore, our application enables the user to perform his task without the need to write a script or program or even without any knowledge of computer programming. The Mashup created will help in the decision making process, which is the prima facie requirement for success in corporate world.**

***Index Terms*— Web Data Extraction; Making Mashup; Mashup Stages; HTML; XML; DOM tree**

I. INTRODUCTION

Today’s web content is extremely diversified and abundant. Searching for the required information out of this ocean of data is a real difficult task.

We develop a toolkit wherein users can use visual interface to design mash-ups. We will be extracting data from the Web. Web data extraction is the process of extracting structured information from unstructured or semi-structured web data sources. Web Extraction is also referred as Web Data Mining or Web Scraping.

Our software will first extract the pre- targeted web pages containing the desired data with the help of Robots and web crawlers which give instructions on what to search and what for. When it searches pages from a website, it will also follow any links that might lead to other relevant content. Linked pages are usually similar in content. Wrappers will help in identifying the targeted data.

Data on the web is in unstructured HTML format, which is mainly presentation-oriented and not suited for database applications.

Manuscript received December 7, 2011. P.A.Chaudhari is with Gokhale Education Society’s COE, Nashik,India. (Contact no: 9822926437 email id: poonam.c@india.com)
R. L. Paikrao is with Amrutvahini COE, Sangamner, India. (Contact no: 9823403157 email id: rahul.paikrao@gmail.com)

This is not really suited for database applications. Secondly, querying data in HTML contents incurs high cost and time.

To overcome these drawbacks, unstructured HTML documents need to be converted into a structured format, like XML. XML separates data structure from layout and provides more suitable data representation.

The Document Object Modeling Tree is constructed using a HTML parser. This tree is used for pattern and structure matching. Further, the information needs to be transformed. This is done using filtering, cleaning, transforming and integrating. The software uses scheduling techniques so that the web pages can be used repetitively. The result can be outputted to Database Management system or to Data warehouses for further mining.

II. ARCHITECTURE

Figure 1 shows the Web Data Extraction system [1].
This system comprises of three tightly connected components:

1) *The web*: contains pages with information of interest
2) *The target Application:* to which the extracted and refined data will be ultimately delivered
3) *The user:* who will interactively design the wrapper.

A. *The Major Modules contain:*

1) Web Interaction: specification of URL and marking of data

2) Wrapper Generation: program (ROBOT) that will download the required page, parse it using HTML Parser and generate tree from parsed output.

3) Extraction: comparing generated tree with saved one to extract updated data from current web page available.

4) Scheduling: repeated execution of above

5) Data Transformation: EMail Updates, SMS Updates, Data Warehousing Using XML, Tabular View.

6) Generation of Mashups: Deliverable APIs or libraries that can be used to extract required information.

B. *Mashup* [2]

A mashup is a web application that combines information and functionality from a number of different websites into an integrated view. A mashup consists of an API, mashup sie and the client's Web browser.

Xtractorz [3] is designed to implement web data extraction and making a Mashup, it will make it easy for the user to extract the relevant data from various web pages without having programming skill.

C. Robot Creation Steps

When we create a robot for the extraction purpose, we need to specify:
a) URL
b) Data that is important to us

Steps included are as follows:
1. *Parsing and tree generation*

Once we specify the important data on that web page, the robot will parse the page into tokens. The parser attempts to balance opening tags with ending tags to present the structure of the page. The output of the parser is a Document Object Model tree.

2. Locating data

Once tree is generated, it will traverse through the tree using Breadth First search for locating the required data.

3. *Save the path*

Once the data is located, the robot will save the path pattern that it took to reach the data in the tree. There after all the robot needs to do is to save the URL and path that it took initially to reach the data.

4. *Recursive search in targeted Web Pages*

When we need to extract the data from an updated web page, the robot will parse it again and traverse the same path that it had in stored to reach the appropriate data.

In order to do this it must compare two paths and hence the need for algorithm.

Figure 2 shows the overview of Xtractorz system

III. THE SYSTEM

Figure 3 shows the system which consists of two major algorithms:

1) DOM Tree Creation
2) Tree matching algorithm

A. Algorithm for DOM Tree [4]

```
DomTree($tag,$CodeHtml,$Parent,$Index)
{
    // Parsing $CodeHtml
   $ResultParsing=ParsingCode($Tag,$CodeHtml);
    // To Stop Recursive Condition
    If (NodeLeaf($HasilParsing))
   {
          Exit;
    }

   Else
   {
        // Find Tag Child
        $TagChild=Array();
        $TagChild=FindTagChild($ResultParsing)
        for (i=0;i<count($TagChild),i++)
          {
                DomTree($TagChild[$i],$ResultParsing,
                $Index,$Index++)
           }
      }
  }
```

The complexity of this algorithm is *O (n1n2)*
Where n1 = number of nodes in first tree
n2 = number of nodes in second tree

B. Tree matching algorithm [5]

Part I: To initialize the pattern structure

```
index:=1;
base:=1;
//Initialize the content and nextnode fields of pattern;
for(i:=1;i≠0;i:=P[i].nextnode)
{
      //initialize back field of pattern nodes//
      if (P[i].content = P[1].content)
             P[i].back:=0
      else
             P[i].back := 1;
}
```

Part II: Used for preprocessing

```
for(i:=2;i≠0;I :=P[i].nextnode)
{
   if (P[i].content = P[1].content)
   {
      base :=1;
      Descend(i,1);
      //find the most repeated region//
   }
}
```

Part III: is the Descend procedure used in part II

```
Descend(i,index)
//compare for descendants, if the corresponding//
//nodes' contents are the same, then continue//
//comparing to find the most repeated region//
{
   index:=P[index].nextnode;
   temp:= [log index];
   k := 2^temp x (base-1)+index;
   //compute the corresponding node//
      if (P[k].content ≠null)
       if (P[k].content ≠P[index].content)
         if P[k].back<index
              P[k].back := index
   //find the most repeated region//
    else Descend(k,index);
   //continue comparing next node//
}
```

Time complexity of this algorithm is *O (m x log m)*
Where m = number of nodes in the pattern
This algorithm can be implemented in Java.

IV. PERFORMANCE EVALUATION

Evaluation results for implementing visual web extraction and making a mashup between RoboMaker, Karma and Xtractorz.

The performance measuring criteria are:

1) Number of steps required for building a mashup
2) Precision of extracted data

Figure 4 shows the result evaluation, X axis maps Mashup stages and Y axis shows the number of steps.

V. APPLICATIONS

The database created can be input to data mining tools, data warehouse, Email server or SMS server.
For example:
1) The power consumption of each household can be calculated and recorded. The demand ratio can be studied to increase the supply accordingly, like in summer seasons when the power consumption is at its maximum due to heavy use of fans/cooler/air conditioner, water motor, etc.

2) The annual rainfall can be measured and compared with the database to analyze the water level in dam and chances of flood, accordingly an alarm through SMS or through email can be sent to the respective authorities.

3) A particular stock can be studied for its 56 week high and low through the database to predict its current value.

4) Mashup can be useful in weather forecast or to alarm about a storm or heavy rains based on the historical data and patterns available.

5) Sales are based on the economical growth of a country. If the country's economy is booming, sales of luxury commodities increase. Analysis of Gross Domestic Products (GDP) rate can help to forecast the economy status.

VI. CONCLUSION AND FUTURE WORK

It can be concluded that the Xtractorz can give a positive contribution in terms of algorithm technique to web data extraction and building Mashup stages. A unified GUI framework will ease the users to perform the extraction of web data without having to have the expertise in creating a computer program.

The available features in the Xtractorz GUI can be enhanced with JQuery facility and the NLP (Natural Language Processing) to further collaborate semantic meaning of the attributes or the table contents (cells) which are extracted. Semantic modeling can be integrated with the concept of Ontology as an alternative technique of DOM tree modeling, in order to find the best solution to implement web table extraction process and building a Mashup.

REFERENCES

[1] Robert Baumgartner , Wolfgang Gatterbauer, "Web Data Extraction", 2010
[2] Journal of Computer Science 7 (2): 129-142, 2011 ISSN 1549-3636 © 2011 Science Publications " Proposing the new Algorithm and Technique Development for Integrating Web Table Extraction and Building a Mashup"B. Smith, "An approach to graphs of linear forms (Unpublished work style)," unpublished.
[3] Rudy AG. Gultom, Riri Fitri Sari, "Implementing Web Data Extraction and Making Mashup with Xtractorz", 978-1-4244-4791-6/10/$25.00_c 2010 IEEE
[4] Jer Lang Hong, Fariza Fauzi, "Tree Wrap-data Extraction Using Tree Matching Algorithm", February 2010
[5] Majlesi Journal of Electrical Engineering Vol. 4, No. 2, June 2010-43,"Tree Wrap-data Extraction Using Tree Matching Algorithm"

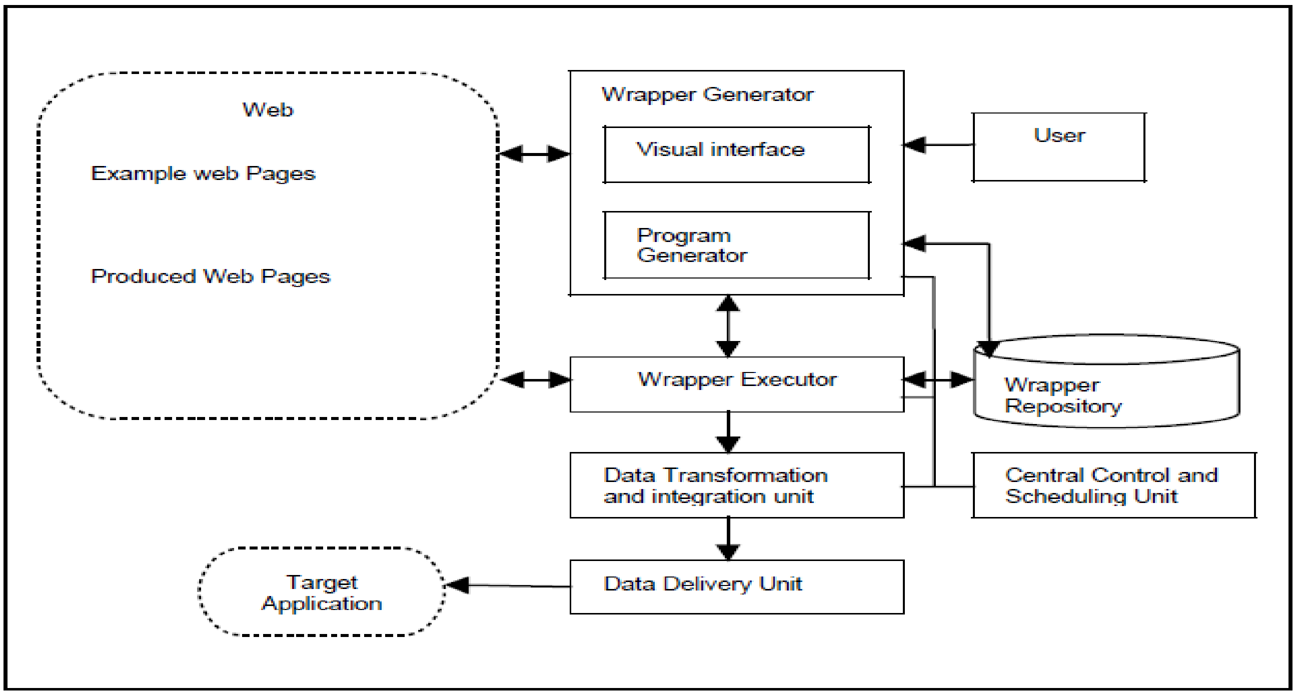


Figure 1: Web Data Extraction System

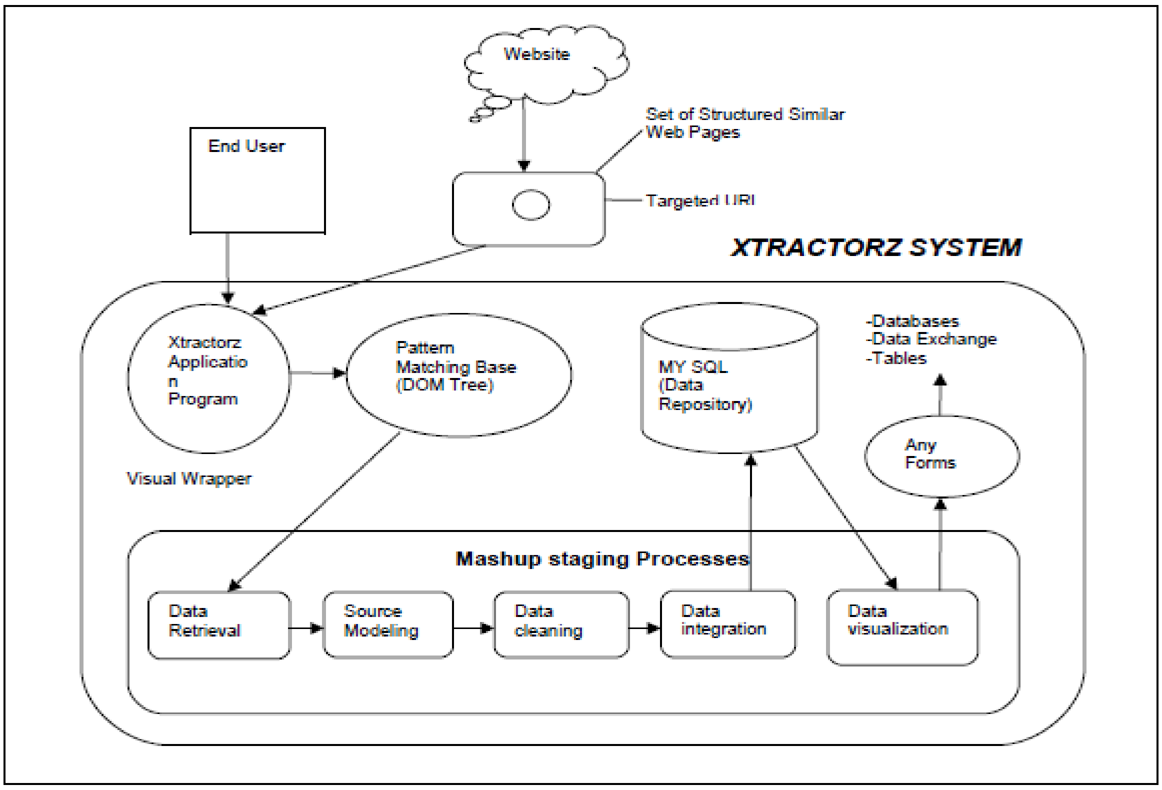
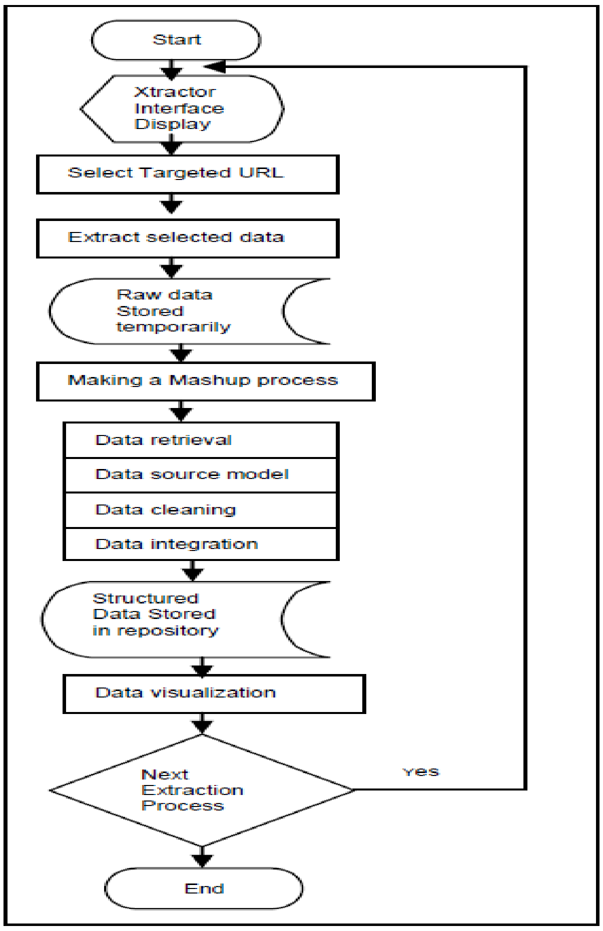


Figure 2: Overview of Xtractorz system



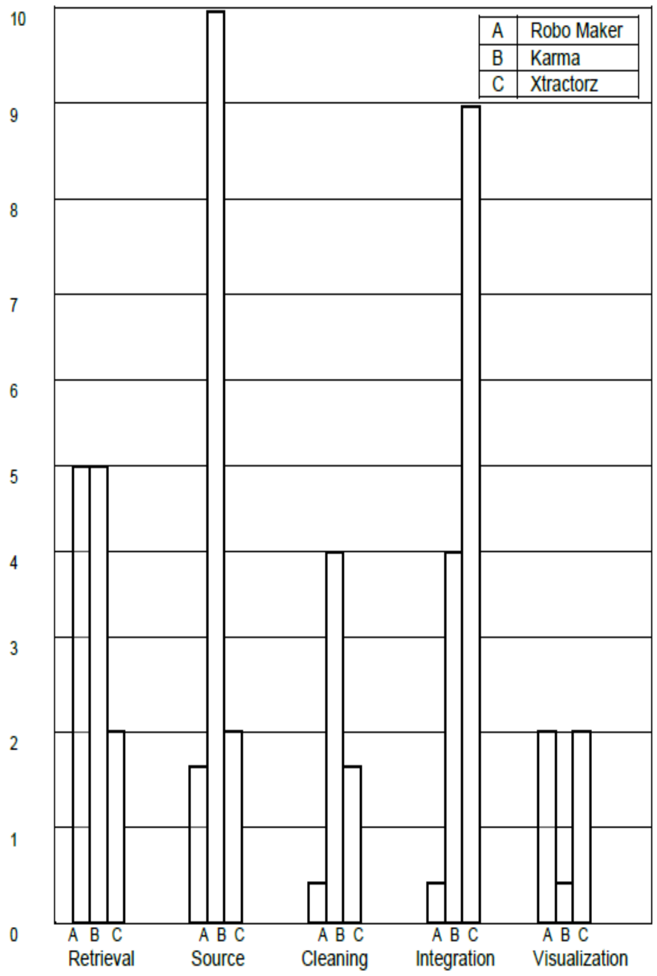


Figure 3: The System

Figure 4: Result Evaluation