IEEE Article Data Extraction from Internet.

Nam Pham*, Bogdan M. Wilamowski**
*Auburn University, Department of Electrical and Computer Engineering, Auburn AL, U.S.A
Email: nguyehu, wilambm@auburn.edu

***Abstract:*-Article data extraction from internet is a way to download and extract the required data automatically from web servers. In this paper, we present a method called the Internet Robot to extract the data directly from a web server by using Perl scripting language with the powerful regular expressions. The regular expressions are widely used in this method to reduce the complexity of the program code as well as increase up the downloading and extracting speed. The Internet Robot in this paper is a process of three steps: data collection, data filtering and processing, data presentation. The final result of this process will be the html files- with all required data in the format as Fig. 1- presented under different links of a webpage as Fig. 5. The accuracy and speed make this method become unique in processing and extracting data not only from the internet but also from an available database.**

I. INTRODUCTION.

With a tremendous growth in available information to the masses [1] the question is how users can search and extract the useful information they need in the shortest amount of time. In other words, making use of consolidated information requires such substantial efforts since the web pages are generated for visualization not for data exchange [6]. This requires methods developed to optimize users' searching process. This paper introduces a method known as the Internet Robot (IR) to process and extract data from the web servers by using Perl scripting language. The Internet Robot can be understood quickly as a special program that serves for a certain purpose of users particularly in processing and extracting data automatically from a web server with a structured template. The programs that perform the task of Information Extracting (IE) are referred to as Extractors and Wrapper [8]. This method doesn't use browsers to handle the web but it does that by using modules such as *LWP* (the extensive libwww-perl library) [2]. This aspect turns the Internet Robot into an effective solution to extract data with such an accelerated speed [1].

The executing procedure of the Internet Robot can be divided into three steps. 1: *Data collection*: to extract all information from web pages into database to analyze. 2: *Data filtering and processing*: the useful information needs to be extracted from the database with a random mass of information. 3: *Data presentation*: all extracted information is processed and sorted in the format which is convenient for users.

To have a closer look about how the Internet Robot really works, this paper will describe more details with a typical application how it extracts all information about authors' name, titles, volume numbers, issue numbers, page numbers, issue dates…automatically from IEEE Xplore and rearranges them in the typical format that readers can search and read papers in the fastest way as well as know all details about authors, dates, citations...in the easiest way. Overview about the Internet Robot is given in section II, section III will explain all details how to execute the Internet Robot to download and extract useful data with illustrating figures from servers of IEEE Xplore, particularly IEEE Transactions on Industrial Electronics. Section IV will be the conclusion.

II. OVERVIEW ABOUT THE INTERNET ROBOT.

Execution of the Internet Robot is very simple. Programmers only need Perl scripting language which is an interpreted language, is parsed and executed at runtime instead of being compiled into binary form and then run [2]. Moreover, this is the open-source software for users with the standard modules which also comes with Perl [3]. Like the built-in functions, these modules provide users with hundreds of prewritten resources. A module is made up of Perl code written in a way to conform to certain conventions so users can access that code from their program. Perl modules provide users with a great deal of prewritten code and are stored in files with extension *.pm*. Users can load such modules into their code by using the *use* statement [2].

1. Perl scripting language.

Perl which stands for "Practical Extraction and Report Language" is a way to make the report processing easier. It is a great language for doing data manipulation tasks. It is fast, portable and accessible. Perl is the programming language which excels at handling textual information. It has been used with good success for systems administration tasks on Unix systems, acting as glue between different computer systems, World Wide Web, CGI programming, as well as customizing the output from other programs [2].

Perl is the most prominent web programming language because of its text processing features and development in usability, feature, and/or execution speed. Handling

978-1-4244-4113-6/09/$25.00 ©2009 IEEE

HTML forms is made simple by the CGI.pm module, a part of Perl's standard distribution. Perl has a powerful regular expression engine built directly into its syntax. A regular expression is a syntax that increases ease of operations that involve complex string comparisons, selections, replacements and hence, parsing [1].

2. Web browser.

As we said before in this paper, the Internet Robot doesn't use browsers but it plays that role by using modules. These modules have capability to download a web page. In other words, they can emulate as a browser. There are a number of modules supporting programmers to do that in Perl such as *LWP::Simple* or *LWP::UserAgent*…[2] Let's look at a typical following example to download the main FAQ index at CPAN by using *get* function of *LWP::Simple* module and store that web page into a file, name.html.

```
use LWP::Simple;
$add="http://www.cpan.org/doc/FAQs/index.html";
   $content=get("$add");
   open FILEHANDLE,">name.html";
   print FILEHANDLE $content;
   close FILEHANDLE;
```

An address is called into the subroutine &content, this subroutine uses get function of LWP::Simple to copy web source into the variable $content. And then the content of this web page is stored into file name.html. The final result will be a file having the similar content as web page http://www.cpan.org/doc/FAQs/index.html .

In this application, instead of using LWP::Simple to emulate as a browser, we substitute it by the written *Wget.exe* [4] which is available and familiar with all users. GNU Wget.exe is a free software package for retrieving files using HTTP, HTTPS and FTP, the most widely-used Internet protocols. In another words, its main function is to link to a certain webpage with an address input and copy all web sources of that webpage into a file. Because of that reason, Wget.exe is considered as non-interactive. It allows users to disconnect from the system and let Wget.exe finish their job without logging in. This is extremely convenient and time-saving when users have to transfer a lot of data from different webpage links. In contrast, other web browsers always require users' constant presence. Generally, the Wget.exe works almost the same as the browser built from LWP::Simple module. However, it is more powerful because it gives users more options.

Below is the basic structure to execute Wget.exe from the Perl script. *"–O"* means that the documents will not be written to appropriate files but all will be concatenated together and written to a file while *"–q"* turns off Wget.exe output on the prompt screen.

```
$add= address of webpage;
$fname= "filename";
system("wget.exe", "-q", "-O", $fname,$add);
```

By assigning value of variables $add and $fname by an address of a webpage and a name of a file which will be generated respectively and using structure *"system"* as above, the new html file will contain almost content of that web link except Java script. Once the new html file contains all data that users want, they can open and copy this data into an array. From here, users can extract all information they need by using Perl commands, especially the regular expressions which is the power of Perl language and makes it different from other languages.

III. APPLICATION.

1. Goal.

Our goal is to download desired information of papers from 1988 until now from IEEE Transactions on Industrial Electronics with all details about authors' name, titles, volume numbers, page numbers, date issues, content of abstracts, Abstract links and Full Text links which will be active if you are a subscriber of IEEE Xplore as Fig. 1.

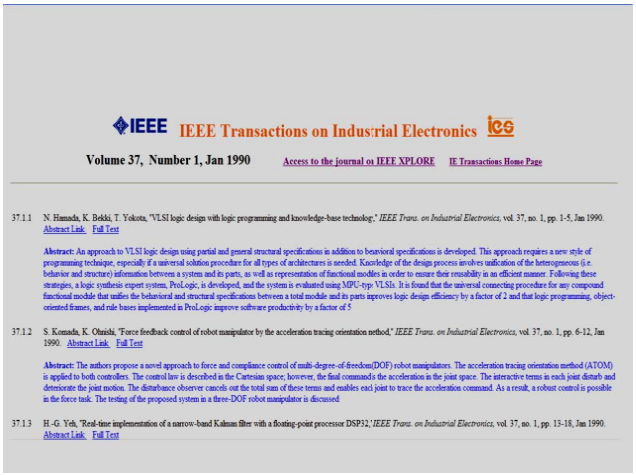

Fig. 1. Format of a html file.

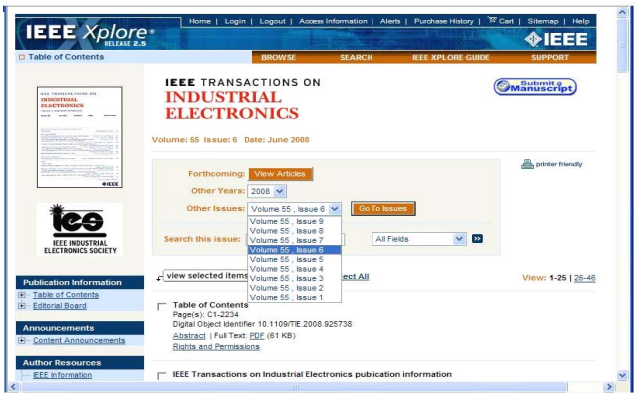

Fig. 2. IEEE Xplore webpage.

With this typical presentation, readers can find out what paper they are interested in the easiest and fastest way by reading information about abstracts, authors,

titles…Once readers want to read or download the full text of paper in PDF format, they can click to the Full Text link without logging their password in IEEE Xplore in case the data base is available. By using the same concept according with Publish and Perish software, we can create a tool to let the users know more information about total number of papers, total number of citations per year, number of citations per paper, the most cited papers and the most downloaded papers.

2. *Execution.*

All magazines or journals' issue level of IEEE have the same format link (OPAC) [7].

"http://ieeexplore.ieee.org/servlet/opac?punumber=X&isvol=Y&isno=Z"

X: code number of a magazine or a journal.

Y: volume number of one year

Z: number of issues in one volume.

In our application, IEEE Transactions on Industrial Electronics has its code number X=41. If users want to know all information about papers of the issue number 6 in the volume 55, they only need to type in: http://ieeexplore.ieee.org/servlet/opac?punumber=41&isvol=55&isno=6 and this link will lead users to papers in that issue (Fig. 2.). Every volume has many different issues and every issue has many papers which consist of 11 subject categories.

For convenience to follow how the Internet Robot works, a flowchart (Fig. 3) is depicted to present all its three steps: data collection, data filtering and processing, data presentation on web.

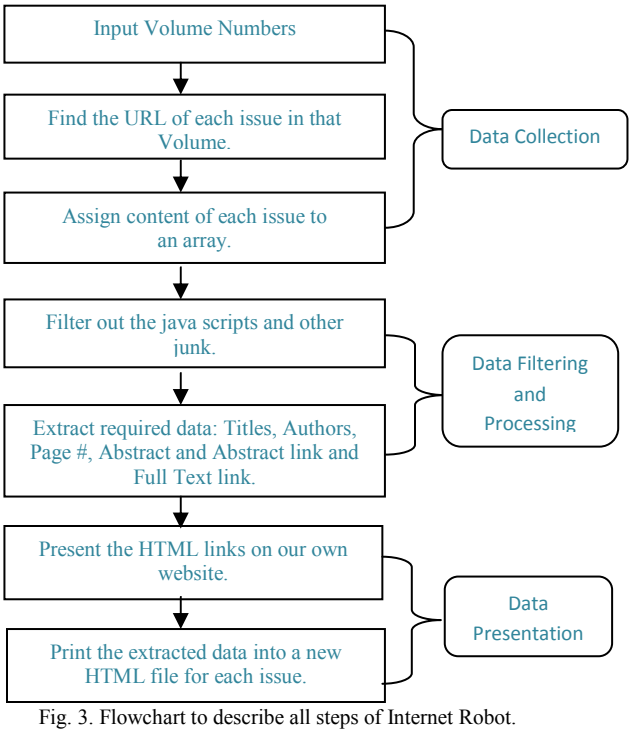

Fig. 3. Flowchart to describe all steps of Internet Robot.

2.1 *Data collection.*

There is one particular feature of Perl that programmers can use outputs of other programs as their input. This is the background to build the control text file. When we input enough data as code number, volume number and issue number…to refer to a web link of IEEE Transactions in the control text file, the input data will be read into an array of Perl program. By assigning the code number, the volume number and the issue number for variables, these variables will receive the same values from the input. This option will give programmers more flexibility without changing the generality of their codes. Fig. 4 is a sample of the control text file in this example.

```
code=41
MAXissue=12
FROMissue=1
FROMyear=1988
TOissue=10
TOyear=2008
startYear=1953
directory=C:/Industrial Electronics on Trans/
journal=IEEE Transactions on Industrial Electronics
shortcut=IEEE Trans. on Industrial Electronics
```

Fig. 4. Control text file.

Moreover, instead of using the standard array as C or C^{++}, programmers can use *hash* in Perl which doesn't depend on the order of input data by searching it text string keys, not numeric indexes. It is very handful when users can easily make some unexpected typing errors. In order to extract all information about titles, pager numbers, authors' name of all papers in this issue, at first the program has to call a sub-routine &sub get_content, this sub-routine will copy source content of the above web page into an array @lines and return it back into the array @get_issue of the main program for example. By using another sub-routine &sub add_issue, all content of the array @get_issue will be copied into ISS_no.html file. However, this html file only contains web source of 25 first papers, it still misses web sources of other papers from 26 to 46 in this particular issue. By using the regular expressions we can fetch links of other papers in the array @get_issue. After these links are fetched, their source contents are copied and then appended into ISS_no.html file by the sub-routine &add_issue. For simplicity, assuming an array @P1 and another array @P2 are two returned values from our fetching process. At this point, @P=@P1.@P2 will contain web sources of whole papers in this issue, no matter how many papers and how many links they are separated into, their content will be copied into the ISS_no.html file automatically. Traditionally, the approach of most researchers for the above example is to

extract URLs from web pages and then use these extracted URLs to retrieve next pages via HTTP request [5].

```
my @get_issue=&get_content($addr);
my $temp=&add_issue(@get_issue);
sub get_content
  {
    $add=$_[0];  # all incoming input
    $fname= "med.htm";
    system("wget.exe", "-q", "-O", $fname,$add);
    $file=$mydir."med.htm";
    open(r1,"<$file");
    @lines = <r1>;
    close(r1);
    unlink($fname); # delete temporary file
    return(@lines);
   }
sub add_issue
   {
    $file=$mydir."ISS_No.htm";
    open(K,">>$file");
    print K "@_";
    close(K);
    return 0;
   }
```

2.2 Data Processing.

In this step, because all raw data that we need is saved into ISS_no.html file, so this file should be opened and copied into an array. The remaining process will base on this array to extract. This array will include in details about authors' name, titles, page numbers…mixed with a lot of junks that stays miscellaneously, disorderly that we don't need to extract. Using regular expressions according with some other built-in commands as *for* loop, *Index*…we can fetch line by line of this array to extract our desired data.

2.3 Extract titles.

From the webpage of IEEE Transactions on Electrical Electronics, we see every paper has all its information arranged orderly from title, authors' name, page number, abstract link and full text link. So whenever content of papers of this issue is copied into an array, its elements will contain all that information in the same order respectively. In other words, the starting point is to search titles first. However, between our desired information there is a lot of undesired information so called junks that we don't need, therefore we have to filter them out.

The array element which contains the title has the following format.

```
<td
class="bodyCopyBlackLargeSpaced"><strong>TITLE</st
rong><br>
```

To extract the title we need to fetch the element which contains the title. This is the unchanged procedure and will be applied through our remaining process. The array element that has data about the title stores some unique key words and different format comparing with other elements, therefore it is the premise to define its position in the array by using the conditional statements and the regular expressions. Once the title is fetched we can use other commands such as index and substring…to discard junky words so called key words above that we don't really need by allocating the starting point and the ending point of a title. The result will be a string having the title of a paper and stored into a variable which will be used later. Particularly in this case, junky words as: <td class="bodyCopyBlack LargeSpaced"> and
 will be avoided.

2.3.1 Extract authors' name and page numbers.

Once the title of a paper is fetched, the next two elements of the array will be two strings that contain information about author's name and page number. This aspect depends on structured template of web pages as well as desired data. However, the elements which store author's name and page numbers have some similar key words as other junky elements in this array, therefore it will be very difficult to fetch this information by using the regular expressions because the returned result at that time will be something different from the one we want. Maybe, the returned result is the last element having the same key words as in the regular expressions. However, the position of the array element containing the title has been defined already in the previous step, so it can be used as a reference to search for other elements of author's name and page numbers. Still repeat the same steps, fetching the element that have information about authors' name and page numbers first and then extracting author's name and page numbers. The results will be two strings, one string of author's name and one string of page numbers and saved into two different variables. This procedure is kind of similar to procedure of extracting titles.

2.3.2 Extract abstracts, Abstract links and Full Text links.

Two strings below are the typical format of array elements that store information about Abstract link and Full Text link, respectively.

```
<a
href="/xpls/abs_all.jsp?isnumber=4505400&arnumber=44
54446&count=43&index=3                    "
class="bodyCopySpaced">Abstract</a>.
```

```
href="/iel5/41/4505400/04454446.pdf?isnumber=4505
400&prod=JNL&arnumber=4454446&arSt=1893&ared=1
909&arAuthor=Levi%2C+E."class="bodyCopySpaced">P
DF</a> (526 KB)
```

To get the Abstract link and the Full Text link of a paper we need to fetch elements in the array that store the Abstract link and the Full Text link [5]. Although these fetched elements store information of the abstract link and the full text link but they are not the full links that can be

used in html of the data presentation part. However, this information is not useless to create the full Abstract link and Full Text link that can really work. First we should filter to get necessary information and then concatenate it with partial links that we create by assigning them into variables $b and $c. For example, assuming the variable $abstract="=4505400&arnumber=4454446&count=43&index=3" is a substring that we extracted from a long string above and then concatenating this string with another newly created variable which is also a string $b="http://ieeexplore.ieee.org/xpls" ($link_abst= $b.$abstract;), the final string will be the abstract link of that paper.

The results of this process will be some respective links like these for Abstract link and Full Text link.

http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?isnumber=4505400&arnumber=4454446&count=43&index=3

http://ieeexplore.ieee.org/Xplore/login.jsp?url=/iel5/41/4505400/04454446.pdf?tp=&isnumber=4505400&arnumber=4454446

When we had the abstract link, we can use a sub-routine to call that address and copy all its content into an array by using Wget.exe. From this array we can fetch the array element storing a full description of paper abstract by another sub-routine &get_abstractspan. Once it has been done, the full description of abstract will be returned and assigned into a variable $each_abst.

```
my @get_abstract= &get_content_ab($link_abst);
my$abst_content= &get_abstractspan(@get_abstract);
$each_abst[$jj]= $abst_content;
```

2.4 Data Presentation.

Our desired data about Author's name, volume number, issue number, page number, abstract link, full text link, abstract description of a paper is fetched, extracted and stored in variables. Combining html with these variables in Perl script, these data will be presented in the favorite format which is easy and convenient for readers. Still fetching remaining elements in ISS_no.htm file, all authors' names, titles, abstracts…of an issue will be extracted, presented and copied into a file name 55_6.htm having similar format as Fig. 1. While the variables $volume_ number and $issue_number are used to track number of an issue in the volume so that the program will automatically update a new name for the new issue, not overwrite on the old issue, and @add_link is an array consists of all information about authors' names, titles… that has been already presented in html format.

```
$volume="$volume_number"."_"."$issue_number";
$file=$mydir."$volume.htm";
        open(Kvolume,">$file");
        print Kvolume "$head <table>";
        print Kvolume "@add_link";
        print Kvolume "</table>";
        close(Kvolume);
```

Expanding this concept and using for loop, all issues of volumes can be extracted at one time. Every issue will be generated into every html file, and all issues in one volume as well as many volumes will be grouped and generated in another html file. The obtained files are then released into the World Wide Web by presenting them as links on a website. A sample website which contains the links of all desired data can be accessed from

http://tie.ieee-ies.org/tie/abs/index.htm (Fig. 5.)

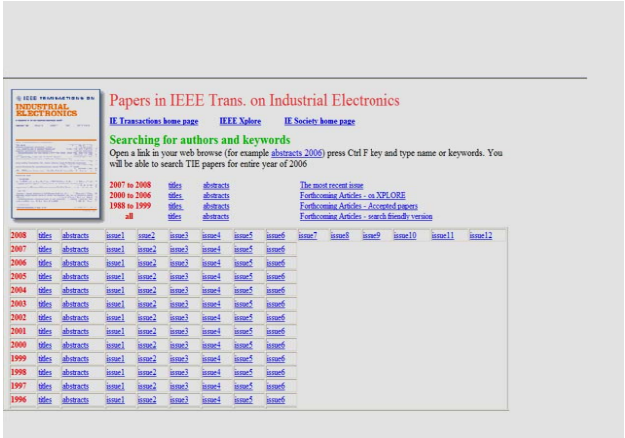

Fig. 5. Resultant webpage.

2.5 Other applications.

A. *"Publish and Perish"* is also a software written in Perl script to extract number of citations of papers. By inputting enough information about journal's name, year, field… and running, this software will generate a text file including in authors' name, titles, number of citations… of papers. If this text file is saved and used as an input of our another program, this program will compare each paper of each volume that we downloaded from IEEE Xplore with papers from the text file to create a new html file with number of citations added (Fig. 6.)

Fig. 6. Volume papers with a number of citations.

B. Using the Internet Robot, we can download forthcoming papers weekly or monthly and save into two different files. Nevertheless, there are some old papers removed and some new papers added. Therefore, how we

know which papers were removed, which papers are still and which papers have just been added. By comparing these two files, the Internet Robot will generate new three files containing those papers with three different states. For example, there are 101 forthcoming papers on September 21th and 94 forthcoming papers on October 25th, after comparison, the result shows that there are 21 old papers removed (Fig. 8), 14 papers have just been added (Fig. 7.) and 80 common papers are still (Fig. 9.). In this case because the Internet Robot doesn't have to access to extract data directly from IEEE Xplore but access directly to available database, so its speed is accelerated extremely faster.

IEEE Transactions on Industrial Electronics

14 new forthcoming articles as of Oct 25 2008

Access to the journal on IEEE XPLORE IE Transactions Home Page

1 H. Tanaka, K. Ohnishi, H. Nishi, T. Kawai, Y. Morikawa, S. Ozawa, T. Furukawa, "Implementation of Bilateral Control System Based on Acceleration Control Using FPGA for Multi-DOF Haptic Endoscopic Surgery Robot," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

2 K. Zhou, D. Wang, B. Zhang, Y. Wang, "Plug-In Dual Mode Structure Repetitive Controller for CVCF PWM Inverters," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

3 B. Wang, X. Yu, X. Li, "ZOH Discretization Effect on High-Order Sliding Mode Control Systems," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

4 K. Erbatur, O. Kurt, "Natural ZMP Trajectories for Biped Robot Reference Generation," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

5 C. Kral, F. Pirker, G. Pascoli, H. Kapeller, "Robust Rotor Fault Detection by Means of the Vienna Monitoring Method and a Parameter Tracking Technique," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

6 T. Chen, L. Chen, L. Sun, X. Li, "Design and Fabrication of a Four-Arms Structure MEMS Gripper," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

7 J.P. V. S. Cunha, R.R. Costa, L. Hsu, "Design of First Order Approximation Filters for Sliding Mode Control of Uncertain Systems," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

8 J. Zaragoza, J. Pou, S. Ceballos, E. Robles, P. Ibanez, J.L. Villate, "A Comprehensive Study of a Hybrid Modulation Technique for the Neutral-Point-Clamped Converter," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

Fig. 7. New papers.

IEEE Transactions on Industrial Electronics

21 old forthcoming articles as of Sept 22 2008

Access to the journal on IEEE XPLORE IE Transactions Home Page

1 J. Nerg, M. Rilla, J. Pyrhonen, "Thermal Analysis of Radial Flux Electrical Machines with a High Power Density," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

2 Y.Y. Motai, A.A. Kosaka, "Hand-Eye Calibration Applied to Viewpoint Selection for Robotic Vision," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

3 R.R.-J. Wai, C.C.-Y. Lin, C.C.-Y. Lin, R.R.-Y. Dunn, Y.Y.-R. Chang, "High-Efficiency Power Conversion System for KW-Level Stand-Alone Generation Unit With Low Input Voltage," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

4 C.C.-Y. Wu, T.T.-F. Wu, J.J.-R. Tsai, Y.Y.-M. Chen, C.C.-C. Chen, "Multi-String LED Backlight Driving System for LCD Panels With Color Sequential Display and Area Control," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

5 A.A.G. Espinosa, J.J.-R. R. Ruiz, J.J. Cusido, X.X.A. Morera, "Sensorless Control and Fault Diagnosis of Electromechanical Contactors," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

6 A.A. Di Gerlando, G.G. Foglia, R.R. Perini, "Permanent Magnet Machines for Modulated Damping of Seismic Vibrations: Electrical and Thermal Modelling," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

7 B.B. Castillo-Toledo, S.S. Di Gennaro, A.A. G. Loukianov, J.J. Rivera, "Hybrid Control of Induction Motors via Sampled Closed Representation," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

Fig. 8. Old papers.

IEEE Transactions on Industrial Electronics

80 common forthcoming articles as of Oct 25 2008

Access to the journal on IEEE XPLORE IE Transactions Home Page

1 M. Riera-Guasp, J. Antonio-Daviu, M. Pineda-Sanchez, R. Puche-Panadero, J. Perez-Cruz, "A General Approach for the Transient Detection of Slip-Dependant Faulty Components Based on the Discrete Wavelet Transform," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

2 L. Palma, M.H. Todorovic, P. Enjeti, "Analysis of Common Mode Voltage in Utility Interactive Fuel Cell Power Conditioners," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

3 K. Basu, J.S. Prasad, G. Narayanan, "Minimization of Torque Ripple in PWM AC Drives," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

4 H.H.Z. Jin, J.J. M. Lee, "An RMRAC Current Regulator for Permanent Magnet Synchronous Motor Based on Statistical Model Interpretation," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

5 H.H. Choi, "Sliding Mode Output Feedback Control Design," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

6 M. Park, D. Chwa, S. Hong, "Anti-Sway Tracking Control of Overhead Cranes With System Uncertainty and Actuator Nonlinearity Using an Adaptive Fuzzy Sliding Mode Control," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

7 M.M. Zamora, H.H. Wu, M.M. P. Henry, "An FPGA Implementation of Frequency Output," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

8 W. Zhou, T.G. Habetler, R.G. Harley, "Bearing Fault Detection via Stator Current Noise Cancellation and Statistical Control," *IEEE Trans. on Industrial Electronics*, vol. 55, no. ?, pp. 1-1, . 2008. Abstract Link Full Text

Fig. 9. Common papers.

IV. CONCLUSION.

This method of data extraction, the Internet Robot, optimizes data processing and makes it become a unique tool in extracting data from webpage. Perl script with regular expressions and web browsers increases the speed of data extracting as well as the accuracy. The Internet Robot is customized according to the required data and the format of data that users desire. Up to this point, the Internet Robot which we have developed can update our webpage (Fig. 5) automatically and download all information of journal and magazine papers as well as conference paper from IEEE Xplore.

REFERENCES.

[1] S. Neeli, K. Govindasamy, B.M. Wilamowski, A. Malinowski, " Auto Data Mining from Web Servers Using Perl Script", *International Conference on Intelligent Engineering System 2008 (INES 2008)*, pp. 191-196, Feb 25-29, 2008.

[2] Steven Holzner, " PERL, Black book", Edition 1999.

[3] http://www.cpan.org/modules/01modules.index.html.

[4] http://www.gnu.org/software/wget/manual/wget.html.

[5] I-Chen Wu, Jui-Yuan Su, Loon-Been Chen," A Web Data Extraction Description Language and Its Implementation", *29th Annual International conference on Computer Software and Applications Conference 2005 (COMPSAC 2005)* , vol2, pp. 293-298, July 25-28, 2005.

[6] M. Keyed, Chia-Hui Chang, K, Shaalan, M.R. Girgis, " FiVa Tech: Page-Level Data Extraction from Template pages", *7th IEEE International Conference on Data Mining Workshops 2007 (ICDM Workshops 2007)*, pp.15-20, Oct 28-31, 2007.

[7] http://ieeexplore.ieee.org/xpl/opac.jsp.

[8] C.H. Chang, M. Kayed, R. Girgis, K.F. Shaalan, " A Survey of Web Information Extraction Systems", *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp.1411-1428, Oct 2006.