A&QT-R 2002 (THETA 13)
International Conference on Automation, Quality and Testing, Robotics
May 23-25, 2002, Cluj-Napoca, Romania

INTERNET, BROWSERS AND LOCAL DATABASES

Jäntschi Lorentz **Avram Dana**

Technical University of Cluj-Napoca
Department of Chemistry

"Babeş-Bolyai" University of Cluj-Napoca
Department of Informatics

Abstract. In present work propose an automating technique for manipulating local dates such that the dates will become available by TCP/IP protocol and to serve for placing on a internet server.

A Delphi program was build for automating generate of a web page that it contain the links for shared data files and folders.

Key-words: Delphi programming, local databases, hypertext document processing

1. Introduction

In many situations, the activity of internet searching are finalized through downloading a numerous files, with immediate or future relevance and the user is confronted with problem of organizing the information for future reusing.

Proportionally with the quantity of information, stored in files, the activity of re-finding is more difficult to make.

Diversity of file types, information categories and original sources of downloading consume time to organize, access and find a specific information, even if the information are stored in a single computer, and much more, if the information is stored in many computers, in a local network [1].

The problem of finding information is solved separately by many software producers. Thus, Adobe [2], starting with Acrobat 3.0 version it creates a Index program, that browse local PDF files and organize them for findings. More, starting with 5.0 versions, this facility is integrated in Acrobat Reader Program. Microsoft [3] it built many programs or routines for finding information in specific files: text files, word files, excel worksheets and power point presentations. Small firms, such that C. Ghisler & Co. (which produce Windows Commander) usually include in her programs routines for browsing archives and directories.

For research activities is very practice to store the information in specific directories and later, using the information stored the researcher compare own results with results obtained by others, or use the others results to simplify own research activity [4].

Repeated browsing of stored dates lead to idea that exploiting of stored information is more efficiency that exist a program that manipulate the dates to create a summary [5].

Leading this idea, a computer program was build.

2. Program Specifications

The program, named HtmlDatabase [6] generates an html source that contain complete list with links of files stored in a folder and all subfolders.

A case study of efficiency in manipulating of information was showed that is more efficient to store information in a structure like in Fig. 1:

```
Root Data Directory (for specific field of information)
        Directory_1 (an publishing house, as example)
                Subdirectory_1_1 (an revue, an volume, as example)
                        File_1_1_1
                        …
        …
        Directory_n
                …
                        …
```

Fig. 1. Directories structure for data processing

Supplementary information about stored information in files is very useful for the user. The program it considers as default this problem. In every subdirectory Subdirectory_X_X the program are looking for files with extension HTM that it consider to contain additional information for files from subdirectory.

A complex routine named SFT search all available HTM files after names of all other files from directory. After the finding, other routine, named SFA, that consider a large variation of HTML formats look for begin and end of information associated with considered file. The specific html instructions are processed and eliminated, and pure textual information remains. This information are afterwards analyzed to find specific parameters such name of publishing house, revue volume number, article page numbers and title, authors, year of apparition, if this dates are available.

Finally, are generating the index named HtmlData.htm with link information from Root Data Directory, with respect to directory structure (
 markers) and for every file with additional information this additional information are included in a specific order: article title, author names, revue name and volume number, publishing house, article page numbers, year of apparition.

From sizing considerate, the using of markers in HtmlData.htm file is limited to strictly necessary. A case study of file size for HtmlData.htm was show that for numerous files, the size grows considerably and this problem can afterwards make difficulties otherwise.

3. Program Design

The program was build in Borland Delphi 5.0 and was designed in two versions: one of them that are using Windows resources and interface, and other those are using Dos32 console resources and interface. Both versions are compiled and tested at numerous cases of directory structures and files in two subversions: fast version, that are placed in Root Data Directory and run automatically without any user interface and dialog user interface version, that require at least name of Root Data Directory.

The sources of program are stored in three files: main program file, named HtmlDatabase.dpr (Delphi project) that uses the units Tree.pas and UEd.pas and include the resource BDHTML.res. Main program generate the HtmlData.htm file and put main markers in them. All other jobs are preloaded by specific routines from units [7].

The UEd.pas unit contains functions that process any available html file for additional information about data files (interface functions are SFA and SFT).

The standards for processing html files are implemented with respect to American Society, EMBO (European Journal of Molecular Biology Online), Springer Verlag, Elsevier Interscience, American Chemical Society and American Institute of Physics html source files. The idea is that all html files from a publishing house frequently respect same standard, and allow automat processing. The exceptions of this rule are also considered and successfully solved.

The Tree.pas unit contains data structures used in program for ordering of links to data files [8]. In implementation of them oriented object programming are used [9]. The source of file is:

```
unit Tree;

interface

type

  plista = ^lista;
  lista = record
    inf : string;
    urm : plista;
  end;

  sirlis = object
    cap : plista;
    cur : plista;
    procedure init;
    procedure initp;
    procedure ad(sa : string);
    function print : string;
    procedure done;
  end;

  parbore = ^arbore;
  arbore = record
    inf : string;
    adr : parbore;
    ast : parbore;
  end;

  inreg = object
    pa : parbore;
    procedure init;
    procedure ad(sa : string);
    procedure SDRD(sd : parbore);
    procedure SRDP(var li : sirlis;
sd : parbore);
    procedure print(var li: sirlis);
    procedure done;
  end;

implementation

uses

  sysutils;

  procedure sirlis.init;
  begin
    cap := nil;
  end;

procedure sirlis.initp;
begin
  cur := cap;
end;

procedure sirlis.ad(sa : string);
var
  urm : plista;
begin
  if sa = '' then exit;
  new(urm);
  urm^.inf := sa;
  urm^.urm := cap;
  cap := urm;
end;

function sirlis.print : string;
begin
  if cur = nil then begin
    print := '';
    exit;
  end;
  print := cur^.inf;
  cur := cur^.urm;
end;

procedure sirlis.done;
var
  urm : plista;
begin
  cur := nil;
  while (cap <> nil) do begin
    urm := cap^.urm;
    dispose(cap);
    cap := urm;
  end;
end;

procedure inreg.init;
begin
  pa := nil;
end;

procedure inreg.ad(sa : string);
var
  newp : parbore;
  ante : parbore;
  dire : integer;
```

```
  begin
    if sa = '' then exit;
    if sa = '.' then exit;
    if sa = '..' then exit;
    ante := pa;
    if pa <> nil then repeat
        dire :=
AnsiCompareText(sa,ante^.inf);
        if dire < 0 then newp :=
ante^.ast;
        if dire > 0 then newp :=
ante^.adr;
        if dire = 0 then exit;
        if newp = nil then break;
        ante := newp;
    until false;
    new(newp);
    newp^.inf := sa;
    newp^.adr := nil;
    newp^.ast := nil;
    if pa = nil then pa := newp else
begin
      if dire < 0 then ante^.ast := newp;
      if dire > 0 then ante^.adr := newp;
    end;
  end;

  procedure inreg.SDRD(sd : parbore);
  begin
    if sd = nil then exit;
    if sd^.adr <> nil then SDRD(sd^.adr);
    if sd^.ast <> nil then SDRD(sd^.ast);
    dispose(sd);
  end;

  procedure inreg.SRDP(var li : sirlis;
sd : parbore);
  begin
    if sd = nil then exit;
    if sd^.adr <> nil then
SRDP(li,sd^.adr);
    li.ad(sd^.inf);
    if sd^.ast <> nil then
SRDP(li,sd^.ast);
  end;

  procedure inreg.print(var li: sirlis);
  begin
    li.init;
    SRDP(li,pa);
    li.initp;
  end;

  procedure inreg.done;
  begin
    SDRD(pa);
  end;
end.
```

Sorting of links to data files are makes simultaneously with input, when are stored in a tree. The print of links to data files are recursively makes in SRDP method of *inreg* object that first visit left sub tree, second visit root and finally visit right sub tree. The destructor SDRD of *inreg* object destroys the elements of tree, and visit first left sub tree, second visit the right tree and finally visit (for destroying) the root.

4. Program Execution

The program is runes by large data sets (about 3000 files) with good results. Execution time is about 10 minutes on a P200 processor based computer. It generate in this case BDHTML.htm file with a size of about 0.6 MB.

The program can be used for a large type of dates. It can organize PDF's, DOC's, PS's, HTM's, and any other types of files that are recognized by Windows system. Pictures captured from execution of program then are looking for PDF files are showed in Fig. 2.

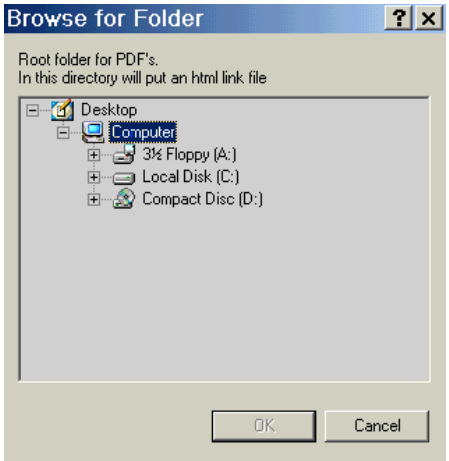

Fig. 2. HtmlDatabase.exe Program Execution

The file BDHTML.htm becomes source for any html interpreter (i.e. Microsoft Internet Explorer, Netscape Navigator). A captured picture of an interpreted file is showed in Fig. 2.

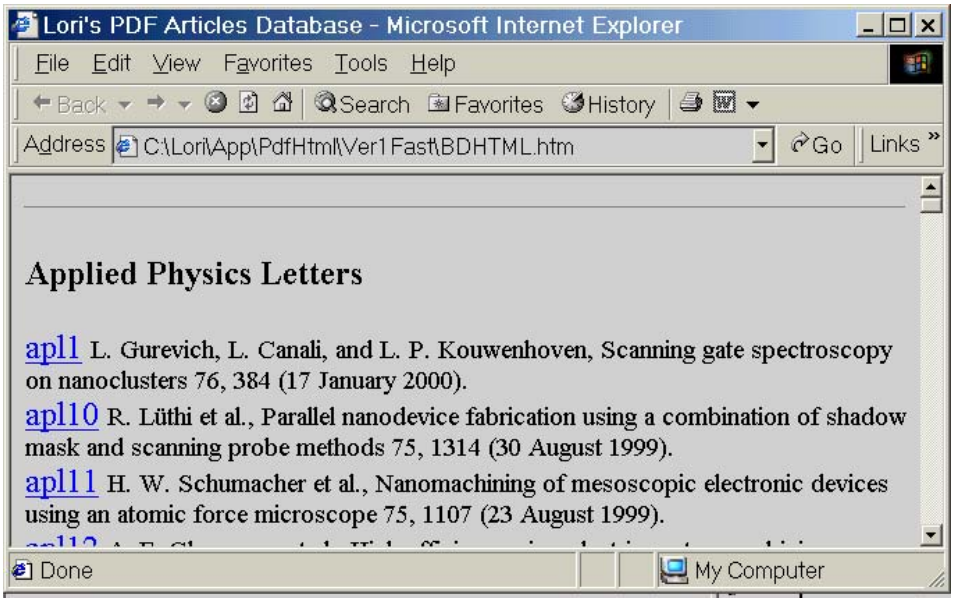

Fig. 3. Examples of output file BDHTML.htm from HtmlDatabase.exe program

The partial source of BDHTML.htm from Fig. 2 is given in Fig. 4.

```
<TITLE>Lori's PDF Articles Database</TITLE><FONT SIZE=4><HEAD><HR><B><BR> Applied
Physics Letters <BR></B>
<BR><A HREF="AmInstPhys\AppPhysLett\apl1.pdf">apl1</A> <FONT SIZE=2>
L. Gurevich, L. Canali, and L. P. Kouwenhoven,
Scanning gate spectroscopy on nanoclusters 76, 384 (17 January 2000).</FONT>
<BR><A HREF="AmInstPhys\AppPhysLett\apl10.pdf">apl10</A> <FONT SIZE=2>
R. L&uuml;thi et al.,
Parallel nanodevice fabrication using a combination of shadow mask and scanning
probe methods 75, 1314 (30 August 1999).</FONT>
<BR><A HREF="AmInstPhys\AppPhysLett\apl11.pdf">apl11</A> <FONT SIZE=2>
H. W. Schumacher et al.,
Nanomachining of mesoscopic electronic devices using an atomic force microscope
75, 1107 (23 August 1999).</FONT>
…
<BR><HR></FONT></HEAD><BODY></BODY>
```

Fig. 4. Examples of output file source BDHTML.htm from Fig. 3

5. Conclusions

The program using it demonstrates his efficiency. The speed of processing is superior to Adobe indexer even that the output file it contain few information comparative with Adobe indexer. The efficiency is augmented through fact that in few cases is necessary to browse entire article to find needed information, and in almost all cases the findings begins with search by subject, article title, authors and publishing house.

The second level of search is assured by html navigator itself, through direct link to requested file, when navigator launch in execution specific program for file (Adobe Acrobat Reader, Ghost View, Microsoft Word, Microsoft Notepad or Wordpad).

Other advantage of this program consist in simplicity of using, and him efficiency depend only by the performance of system that are used.

The program allow to organize downloaded information with same efficiency with it organize personal user information, and, finally, it permit to transfer resulted BDHTML.htm file together with data directory and file structure on a Internet server, for publishing information.

Automat generating of BDHTML.htm file make publishing file activity more easily and consequently with respect to standard specifications [10].

Program also consider and eliminates file repetitions that confer to output file an increased consistency [11].

References

[1] Márton Nagy, Suresh Singh (1998), Multicast scheduling algorithms in mobile networks, *Cluster Computing*, 1, 177-185.
[2] Adobe Systems Incorporated (2001), ***, http://www.adobe.com.
[3] Microsoft Corporation (2001), ***, http://www.microsoft.com.
[4] Jackson State University (1997), Sixth Conference on Current Trends on Computational Chemistry, ***, Vicksburg, Missisipi, Nov. 7-8, 2-178.
[5] Nisbett R. E., Fong G. F., Lechman D. R., Cheng P. W. (1987), Teaching Reasoning, *Science*, 238, 625-631.
[6] Lorentz Jäntschi (2001), HtmlDatabase.exe, *http://www.utcluj.ro/facsim/chem/WebPage/* HtmlData.htm.
[7] D. Knuth (1974), Computer Programming Guide. Fundamental Algorithms (in Romanian), Technical Publishing House, Bucharest.
[8] M. Lerman, R. I. Soare (1980), d-Simple Sets, Small Sets and Degree Classes, *Pacific J. of Math.*, 87-1, 135-155.
[9] M. Frențiu, B. Pârv (1994), Elaborating of Programs: Modern Techniques, Promedia Publishing House, Cluj-Napoca.
[10] L. Jäntschi (2002), Some Aspects of Vertex Equivalence in Graphs, *Australian & New Zeeland Journal of Statistics*, submitted.
[11] Tripos Associates (2000), Unity Program for SIMCA (Soft Independent Modeling Class Analogy), ***, St. Louis, MO.