

# Building the corpus used in the BiRD Project

Richard Evans  
School of Humanities, Languages, and Social Sciences  
University of Wolverhampton  
Stafford Street  
Wolverhampton WV1 1SB, UK  
{R.J.Evans@wlv.ac.uk}

March 16, 2004

## Abstract

This document describes the properties and use of programs used to build the corpus in the BiRD project.

## 1 Introduction

## 2 The programs

### 2.1 `collect_corpora_messages.pl`

This program is called with the name of one of the corpora archive pages in the ARGV[0] position. These archive pages are a list (for a given time period) containing the links to corpora messages stored on the internet. This program downloads the documents at those link locations. It also adds information to the file *meta\_information*. This information consists of the original URL of the message, the download location, the download date, the original HTML page creation date, and a list of the URLs that appear in the message linking it to other pages “to links”.

INPUT are different web pages from the *corpora list* archive (e.g. “...CorporaApr2001toJune2001...”). OUTPUT files are named as `{date}.html`.

### 2.2 `new_follow.pl`

This program processes the HTML files downloaded by *collect\_corpora\_messages.pl*. It prints information to *meta\_info\_for\_followed\_links*. It downloads

documents linked to corpora list messages. Each link is followed exactly once (no duplication). For links to documents larger than 3000 characters, following that link and downloading it causes the program to increment a *level* variable. Links to other corpora list messages (indicated by *previous message:* or *next message:* are not acquired). Anchors are deleted from links and relative links are made absolute with respect to the introducing corpora message location (at the first stage or processing the corpora list message). For absolute links, the initial part of the link comprises a root link for subsequent relative links introduced in that linked document. Unless a link is of a particular file type, and unless it has been downloaded before, information is added to *meta\_info\_for\_followed\_links* and the procedure *follow* is called with parameters set for *link*, *root link* (substring of *link*), *hyperlink* text, and *source link* which is the URL of the document that introduced *link*. This procedure is also applied to documents embedded in frames, though in these cases, the parameters passed to *follow* are *link* and *root link*.

Output files are named LINK\_{nn}.html and NF\_{nn}.html. The program does not move the initially downloaded email messages anywhere, i.e. it does not create duplicate corpora list email messages.

The documents pointed to by *link* are downloaded to a temporary disposable file. This temporary file is then processed. If the length of the document exceeds 3000 characters, the variable *level* is incremented. If *follow* has not been applied to the link before, the temporary file is saved to the corpus that is being built.

The file saved to the corpus is then opened and processed. Links appearing in the saved file are identified and de-relativized. While 20 linked have not been processed from the same saved file, if the document contains a link that has a special hyperlink (e.g. document, manual, help, instruct, overview, etc.), and if the *level* variable  $< 4$ , *follow* is called on that special link, with new parameters that relate to the new link. The document obtained from the special link is appended to the source document containing the special link.

In general, for small files of length less than 3000 characters, the file is deleted. The output from this program comprises the *intermediate\_corpus*.

### 2.3 basic\_tagging.pl

This program processes the *corpora\_list* HTML files and consults *meta\_information*. It adds header tags to the files, detailing the XML doc type declaration and location of the DTD file, the title of the message, document reference number, the date of creation and download, lists of meta

tag keywords, and the URLs of previous and following corpora list messages (where they exist). It also tags elements such as headings, email and web link addresses.

## 2.4 basic\_tagging\_linked\_pages.pl

This program processes *intermediate\_corpus* HTML files and consults *meta\_info\_for\_followed\_links*. It produces files in *intermediate\_corpus/basic\_tagged\_corpus/*. It adds header tags to the HTML files, providing a reference number, URL for the document, the created date for the document, the date on which it was downloaded, the title of the document, the URL of the page that provided the link to this document and the hyperlink of that link. It reports the depth level of the document (i.e. distance from a *corpora\_list* message), and also provides meta data in terms of key words and description where present. Headings, web links, and email address citations within the document are tagged. Initially, an XML !DOCTYPE declaration is provided along with the location of the *bird.dtd*.

## 2.5 deleting\_cvs.pl

This program processes basic tagged files in the *jobs* category. It identifies files marked as [E-CV] and deletes them. This program should be run in each *jobs* subdirectory: *html*, *basic\_tagged\_corpus* and *tidied*.

## 2.6 pretidy.pl

This program processes all the files in a directory, converting ascii based bullet points and lists in HTML structures that use <UL> and <LI>. At present, it is used when processing *conferences* and *jobs* email messages. It must run prior to the TIDY\_SCRIPT.pl programs.

## 2.7 {TYPE}\_TIDY\_SCRIPT.pl

This program takes raw documents downloaded from the Internet as input, encodes non-ascii (UTF-8) symbols as HTML codes (necessary as tidy will only process HTML input) and then tidies the HTML tags, correcting structural errors. tidied files will subsequently be converted to XML and this process requires that the HTML encoding is re-converted as UTF-8.

## **2.8 find\_files\_with\_tidy\_errors.pl**

This program processes the output of a `<TYPE>_TIDY_SCRIPT.pl` and prints all the fatal errors encountered by tidy. On the basis of the printed output, the user can address problems (usually unencoded UTF-8 characters).

## **2.9 golden\_{TYPE}\_basic\_tagging.pl**

This program adds header tags to tidied HTML documents and also recognises and tags elements of interest in the document, such as web and email addresses. Header tags include information on the source location of the document, the web page that links to this document, the date of creation and download, etc. Output from this program is converted into XML using another program. This program adds an XML document declaration tag, assuming that the DTD can be found at <http://clg.wlv.ac.uk/projects/BiRD/bird.dtd>.

## **2.10 printing\_similarity\_between\_files.pl**

## **2.11 deleting\_files\_based\_on\_similarity.pl**

## **2.12 validation\_script.pl**

The user should set the directory containing XML files to be validated, the standard error should be written to a file of the form `<FILE>_validation.log`. This file will contain details of any error found in the input files. Such errors are usually comprised of 2 kinds - structural errors and entity based errors (such as the appearance of HTML codes in the XML document). This script is run after a program has converted the basic tagged documents into XML.

# **3 Streams**

## **3.1 corpora\_list**

1. download corpora list archive pages
2. `collect_corpora_messages.pl`
3. `basic_tagging.pl`
4. TBC

## 3.2 intermediate\_corpus

The ordering of 2 and 3 may, in actuality, be swapped. Those programs prepare each document (email, linked page, or specially linked page) for processing by IC\_TIDY\_SCRIPT.pl. IC\_TIDY\_SCRIPT.pl is substantially different from the other TIDY\_SCRIPTs in that it does no encoding of non UTF-8 characters as HTML characters<sup>1</sup>. Instead, it corrects errors in the HTML tags of the intermediate corpus files (e.g. when computer programs are inserted in to pages, they often contain sequences that resemble HTML tags such as while(i;){. These are removed by IC\_TIDY\_SCRIPT.pl).

1. new\_follow.pl
2. convert\_non\_ascii\_to\_html.pl
3. convert\_utf8\_to\_html.pl
4. convert\_non-utf8\_to\_html.pl
5. IC\_TIDY\_SCRIPT.pl
6. basic\_tagging\_linked\_pages.pl
7. TBC

## 3.3 golden/conferences

1. pretidy.pl
2. CONF\_TIDY\_SCRIPT.pl
3. golden\_conf\_basic\_tagging.pl
4. Viktor's program
5. validation\_script.pl

## 3.4 golden/jobs

1. deleting\_cvs.pl
2. pretidy.pl
3. JOBS\_TIDY\_SCRIPT.pl

---

<sup>1</sup>the previous conversion programs perform this task.

4. golden\_jobs\_basic\_tagging.pl
5. Viktor's program
6. validation\_script.pl

### **3.5 golden/software**

1. SOFTWARE\_TIDY\_SCRIPT.pl
2. golden\_software\_basic\_tagging.pl
3. Viktor's program
4. validation\_script.pl

### **3.6 golden/trash**

1. TRASH\_TIDY\_SCRIPT.pl
2. golden\_trash\_basic\_tagging.pl
3. printing\_similarity\_between\_files.pl
4. deleting\_files\_based\_on\_similarity.pl
5. Viktor's program
6. validation\_script.pl

## **4 Acknowledgements**

This research has been funded by the UK Economic & Social Research Council, award number RES-000-23-0010.